PXE Engineering

Intel Architecture Labs

PXE Product Development Kit Instructions

Version 3.0, Build 082

June 21, 2000

Intel Corporation assumes no responsibility for errors or omissions in this guide. Nor does Intel make any commitment to update the information contained herein.

✶ Other product and corporate names may be trademarks of other companies and are used only for explanation and to the owners' benefit, without intent to infringe.

Intel Corporation, 5200 N.E. Elam Young Parkway, Hillsboro, OR 97124-6497

Contents

Document Revision History

.

Version #	Date	Notes
3.0 Build 082	*6/21/2000*	*Update ROM release notes in section 6.*
3.0 Build 079	*4/27/2000*	*Updated file list in 3.4. There are no more Interrupt 18h or monolithic ROM images for BIOS. Added new NIC/LOM EEPROM programming utility.* *Updated ROM and service release notes in section 6.* *Update file list in 3.4 to include PxeReg60.exe.* *Updated instructions for setting MS Dhcp server on Windows 2000.*
3.0 Build 078	*1/21/2000*	*Updated ROM release notes in section 6.* *Updated FUTIL.EXE instructions.*
3.0 Build 077	*12/1/99*	*Updated ROM release notes in section 6.*
3.0 Build 074	*10/29/99*	*Updated LOM client testing and changed LOM client images in section 3.5.1.* *Added scoping/filtering to proxyDHCP server in section 4.3.2.2.* *Improved client packet analysis in section 5.1.4.* *Updated release notes in section 6.*
3.0 Build 072	*7/24/99*	*Production Update*
3.0 Build 071	*6/28/99*	*Production Update* *Updated FUTIL.EXE instructions* *Clarified .NIC/.LOM image usage in section 3.5.* *Added text to 3.5.2.1.3 describing the new BaseCode selection option in the option ROM initialization menu.*
3.0 Build 068	*4/2/99*	*Production Update* *Updated FUTIL.EXE instructions.*
3.0 Build 067	*2/8/99*	*Production Release* *Added PXE error codes and definitions.*

1. PDK License

INTEL PREBOOT EXECUTION ENVIRONMENT (PXE) PRODUCT DEVELOPMENT KIT (PDK)

COPYRIGHT LICENSE AND DISTRIBUTION AGREEMENT

IMPORTANT: READ BEFORE COPYING, INSTALLING OR USING.

BY USING THIS SOFTWARE, YOU ARE AGREEING TO BE BOUND BY THE TERMS OF THIS AGREEMENT. DO NOT USE THIS SOFTWARE UNTIL YOU HAVE CAREFULLY READ AND AGREED TO THE FOLLOWING TERMS AND CONDITIONS. IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, PROMPTLY RETURN THE SOFTWARE FILES AND ANY ACCOMPANYING ITEMS AND DO NOT COPY, INSTALL OR USE THE SOFTWARE. .

IF YOU USE THIS SOFTWARE, YOU WILL BE BOUND BY THE TERMS OF THIS AGREEMENT AS FOLLOWS:

LICENSE: Intel Corporation ("Intel") grants you a non-exclusive, royalty-free, copyright license to the enclosed software program and materials in object code format, ("the Software") subject to the terms herein. No other license is granted except as expressly provided herein. You will not use, copy, modify, rent, distribute, sell nor transfer the Software nor any portion thereof except as expressly provided in this Agreement.

YOU MAY:

1. Install a copy of the Software on one or more computers for internal use only and for the sole purpose of testing PC systems for adherence to the Wired for Management Baseline Specification;
2. Copy the Software solely for backup or archival purposes and for internal transfer to another user in your company who agrees to the terms of this Agreement.

YOU WILL NOT:

1. Sublicense the Software;
2. Reverse engineer, decompile or disassemble the Software;
3. Copy the Software, in whole or in part, except as specifically provided for in this Agreement;
4. Distribute or transfer the Software outside your company.

TRANSFER: You may transfer a copy of the Software to another person in your company (i.e., the company which was originally granted access to this Software by Intel) on the condition that a copy of this Agreement accompanies the transferred copy and the receiving party agrees to the terms of this Agreement.

OWNERSHIP AND COPYRIGHT OF SOFTWARE: Title to the Software and all copies thereof remain with Intel or its vendors. The Software is copyrighted and is protected by United States and international copyright laws. You will not remove the copyright notice from the Software. You agree to prevent any unauthorized copying of the Software. Except as otherwise expressly provided, Intel does not grant any express or implied right to You under Intel patents, copyrights, trademarks or trade secret information.

WARRANTY DISCLAIMER: The Software is provided "AS IS" without warranty of any kind.

INTEL MAKES NO WARRANTIES OF ANY KIND, WHETHER EXPRESS, IMPLIED OR STATUTORY, AND EXPRESSLY DISCLAIMS WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS, AND FITNESS FOR ANY PARTICULAR PURPOSE. INTEL WILL NOT PROVIDE ANY SUPPORT, ASSISTANCE, INSTALLATION, TRAINING OR OTHER SERVICES. INTEL WILL NOT PROVIDE ANY UPDATES, ENHANCEMENTS OR EXTENSIONS.

Intel further does not warrant or assume any responsibility for the accuracy or completeness of the information, text, graphics, links or other items contained in the software. Intel may make changes to the Software or to any products described therein at any time without notice.

LIMITATION OF LIABILITY: IN NO EVENT SHALL INTEL OR ITS VENDORS BE LIABLE FOR ANY INDIRECT, INCIDENTAL, SPECULATIVE, SPECIAL, OR CONSEQUENTIAL DAMAGES OF ANY KIND, INCLUDING BUT NOT LIMITED TO LOSS OF PROFITS, LOSS OF USE, LOSS OF DATA OR INTERRUPTION OF BUSINESS, WHETHER UNDER THIS AGREEMENT OR OTHERWISE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

TERMINATION OF THIS LICENSE: Intel may terminate this Agreement at any time if you are in breach of any of its terms and conditions. Upon termination, you will immediately destroy the Software or return all copies of the Software and documentation to Intel at Intel's sole discretion.

U.S. GOVERNMENT RESTRICTED RIGHTS: The Software and documentation were developed at private expense and are provided with RESTRICTED RIGHTS. Use, duplication or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor. Use of the Software by the Government constitutes acknowledgment of Intel's proprietary right to them.

EXPORT LAWS: You will not export/re-export the Software in violation of the laws, regulations, orders or other restrictions of the U.S. Export Administration Regulations.

APPLICABLE LAW: This Agreement is governed by the laws of the United States and the State of Delaware, including patent and copyright laws, without regard to its conflict of law provisions.

2. Introduction

This PDK includes a test environment and a series of tests to provide functional and stress testing of PXE Boot ROMs and the PXE APIs. It also includes services necessary for the test server.

2.1 Intended Audience

This PDK is intended for:

1. Developer's implementing PXE
   - Boot ROM OEMs
   - Network Interface OEMs
   - Possibly PC System OEMs
2. OEMs who provide non-PXE elements that Wired for Management (WfM) requires for PXE support
   - PC System OEMs
   - BIOS OEMs
3. OEMs who are developing "Boot Servers" to inter-operate with PXE.

2.2 Equipment Needed

The following table is a guide to which items a particular OEM would test for compliance, and what other compliant elements are required to successfully complete the tests. For example, a "Network Interface OEM" would test their implementation of PXE for their NIC. In addition to their test unit (the PXE-enabled NIC), they would also need a WfM-compliant PC system in which to install their NIC for the tests.

OEMs ⇩	Equipment Needed					
	WfM-compliant PC HW, BIOS, and PXE	**WfM-compliant PC HW and BIOS**	**WfM-compliant PXE-enabled NIC**	**WfM-compliant PC HW**	**WfM-compliant BIOS**	**WfM PXE PDK test setup**
BIOS OEMs	NA	NA	Needed in addition to test unit	Needed in addition to test unit	**Test Unit**	Needed in addition to test unit
Boot ROM OEMs	NA	Needed in addition to test unit	**Test Unit**	NA	NA	Needed in addition to test unit
Network Interface OEMs	NA	Needed in addition to test unit	**Test Unit**	NA	NA	Needed in addition to test unit
PC System OEMs	**Test Unit** (if network interface in PC)	**Test Unit** (if no network interface in PC)	Needed in addition to test unit (if no network interface in PC)	NA	NA	Needed in addition to test unit
Boot Server OEMs	Needed (if network interface in PC)	Needed (if no network interface in PC)	Needed (if no network interface in PC)	NA	NA	Needed

In addition to the equipment listed above, at least one test server and a test network are required. To insure proper operation in a wide variety of environments it is strongly recommended that a second round of testing be done that includes routers in the test network.

3. PDK Installation

3.1 Overview

This section explains how to install the PDK and create a PXE test environment.

To install the PDK you will set up one or more Windows* NT* 4.0 or Windows* 2000 servers, execute the PDK setup program (pxe20-pdk.exe), and create DOS diskette image (boot) files.

In addition, you will set up at least one client PC containing a PXE option ROM. (Although multicast TFTP can be used with a single client, multiple clients are needed to observe the full effect of master/slave multicast TFTP.) The PDK contains binaries and programming utilities to install the PXE ROM image into the FLASH memory on Intel EtherExpress PRO PCI NICs.

Note: The test environment requires the use of a DHCP service. You should set up a test network separate from your main network to avoid conflicting with production DHCP servers.

3.2 Set up the Windows NT or 2000 Server(s)

The PXE test environment includes one or more Windows NT or 2000 servers running Microsoft DHCP, PXE, and MTFTP services. (While it is not absolutely required, we suggest you use a dedicated DHCP server and install the PXE and MTFTP services in pairs on one or more additional servers.)

For all Windows servers in your test environment, do the following:

- Install Windows NT Server (4.0, SP3 or later) or Windows 2000 Server on one or more PCs
- Install TCP/IP services
- Assign a static IP address to the server
- ***Make sure that the GUEST account is enabled.***

Note: The PDK requires the Microsoft DOS network client files from the Windows NT 4 Server CD to build the DOSUNDI and APITEST RAMdisk images. The PDK can be installed without the Windows NT 4 Server CD, but some of the DOSUNDI and APITEST functionality will be disabled. The required files can also be provided later when making the DOSUNDI and APITEST RAMdisk images.

3.3 Set up the DHCP Service

One of the servers in your test setup must provide the DHCP service.

To set up this service on Windows NT 4.0 Server or Windows 2000 Server:

- During the OS installation, install the Microsoft DHCP Server.
- Create a valid scope on the DHCP server.
- Depending on the configuration of your routers, you may have to define a value for the Router option tag. This will ensure the client receives a default gateway IP address.
- ***It is strongly recommended that your initial tests be done without any routers.***

3.3.1 PXE Option Tag Configuration on the Microsoft DHCP Service

Note: The instructions in this section are only for Windows NT 4.0. The PXE option tag is configured automatically on Windows 2000.

If the PXE PDK is being installed on the same host server as the DHCP service and you want to use the proxyDHCP service in the PXE PDK, you must add the DHCP Class Identifier (option 60) to the DHCP service. (Do **not** add this tag to the DHCP service if PXE service is **not** installed on the DHCP server.)

This option must be set to **PXEClient**. The purpose of adding this option to the DHCP service is to inform the client that PXE proxyDHCP service is available on the same host as the DHCP service. When a PXE Client receives option 60 by itself from the DHCP service, the client will immediately unicast a request to proxyDHCP on port 4011 to complete the PXE DHCP configuration.

3.3.1.1 DEFINE OPTION 60

- You must define option 60 to the DHCP service as a string, and then define the value of the data portion of the option. The DHCP service will attach the option number (0x3C) and the data length (0x09).

- Start the DHCP Manager

  Start->Programs->AdministrativeTools (Common)->DHCP Manager

- In the panel labeled DHCP Servers, double-click Local Machine to display the DHCP scopes you have previously defined.
- Select any one of the scopes.
- Click on the menu DHCP Options->Default.
- Click the New button to display the Add Option Type dialog box.
- In the Name field enter the text Class ID.
- Make sure the Data Type is set to String and that the Array checkbox is not set.
- In the Identifier field enter 60.
- Click OK.
  The **DHCP Options: Default Values** dialog is displayed.
- Select the entry or option 060 from the Option Name drop-down list.
- The text 060 Class ID is displayed in the Option Name box.
- Select the button Value>>> to display the field.
- Enter the following string into the new field: **PXEClient**
- Click OK to return to the DHCP Options: Default Values dialog.
- Review the values displayed in the Value group. If corrections are needed, select the Edit Array button and make the corrections.
- Click OK to return to the DHCP Manager.

3.3.1.2 ASSIGN OPTION 60 TO THE SCOPES

Now assign option 60 to one or more scopes. The easiest way to do this is to assign the option globally.

- With one of the scopes highlighted, select the menu DHCP Options->Global to display the DHCP Options: Global dialog box.
- Scroll the Unused Options list to locate the option 060 Class ID. Highlight the option then click the Add button to add the option to the list of Active Options.
- Click OK.

3.3.1.3 TEST THE OPTION

The option is now assigned to all of your scopes. At this point you should test that the option is being correctly sent to the client by performing a client boot while a packet sniffer captures the packets.

- Bring up a DHCP client on the test network (such as an NT Workstation with DHCP enabled).

- Analyze the DHCP Offer packet provided to be sure these option bytes are present.

  The client should receive the following byte sequence in the options section of the DHCP Offer (all values shown in hex):

  0x3c 0x09 0x50 0x58 0x45 0x43 0x6c 0x69 0x65 0x6e 0x74

  - The first byte is the option number 60.
  - The second byte is the length of the option data, 9 bytes.
  - The remaining bytes are the ASCII codes for the characters “PXEClient”.

3.3.1.4 Windows 2000 Specific

The DHCP service included with Microsoft Windows 2000 Server allows easy configuration from a DOS command prompt using “*netsh.exe*”. To further simplify configuring the DHCP service on Microsoft Windows 2000 Server, the PXE PDK includes a utility "*PxeReg60.exe*" that will automatically configure the DHCP Class Identifier (option 60). Automatic configuration will be done only during installation of PDK and when proxyDhcp is enabled or disabled through PXE Configuration Utility. The Microsoft DHCP service must be running when the PXE PDK is installed or the configuration is changed. If not, then DHCP option 60 needs to be configured manually.

"*PxeReg60.exe*" issues these four "*netsh.exe*" commands:

3.3.1.4.1 netsh.exe dhcp server add optiondef 60 “Class ID” string 0 PXEClient

Adds DHCP option 60, with the default value set to 'PXEClient', to the configurable options list.

3.3.1.4.2 netsh.exe dhcp server delete optiondef 60

Remove DHCP option 60 from the configurable options list.

3.3.1.4.3 netsh.exe dhcp server set optionvalue 60 string PXEClient

Assigns option 60, with a value of ‘PXEClient’, globally to all DHCP service scopes.

3.3.1.4.4 netsh.exe dhcp server delete optionvalue 60

To remove any assigned value from DHCP option 60.

3.4 Install the PXE Services

- Copy the self-extracting executable **pxe20-pdk.exe** to a temporary directory of your choice (“<temp>” in the following listing) on the test server.
- Run **pxe20-pdk.exe** to install the services.

The installation program will ask where the files should be unpacked. You may either accept the default or select another directory. The following table shows the files that will be unpacked in the selected directory <temp>.

<temp>\disk1\setup.exe	- PXE PDK setup utility.
<temp>\disk1\setup.ins	- Setup data files
<temp>\disk1\setup.ini	- Setup data files
<temp>\disk1_inst32i.ex_	- Setup data files
<temp>\disk1_ISDel.exe	- Setup data files
<temp>\disk1_sys1.cab	- Setup data files
<temp>\disk1_sys1.hdr	- Setup data files
<temp>\disk1_user1.cab	- Setup data files
<temp>\disk1_user1.hdr	- Setup data files
<temp>\disk1\data.tag	- Setup data files
<temp>\disk1\data1.cab	- Setup data files
<temp>\disk1\data1.hdr	- Setup data files
<temp>\disk1\lang.dat	- Setup data files
<temp>\disk1\layout.bin	- Setup data files
<temp>\disk1\os.dat	- Setup data files
<temp>\pxe_bin\	- PXE ROMs & utilities directory
<temp>\pxe_bin\e100_m.nic	- Boot ROM image for Intel 82557/82558/82559 based PCI NICs with FLASH (Monolithic - Includes BaseCode)
<temp>\pxe_bin\e100_s.nic	- Boot ROM image for Intel 82557/82558/82559 based PCI NICs with FLASH (Split - No BaseCode)
<temp>\pxe_bin\e100_s.lom	- Boot ROM image for Intel 82557/82558/82559 based PCI LOMs (LAN On Motherboard). This is to be tested in system BIOS. (Split - No BaseCode)
<temp>\pxe_bin\pxebase.nic	- PXE BaseCode ROM image for Intel 82557/82558/82559 based PCI NICs with FLASH. This image is for debugging purposes only and will not work in a system BIOS.
<temp>\pxe_bin\pxebase.lom	- PXE BaseCode ROM image for Intel 82557/82558/82559 based PCI LOMs.
<temp>\pxe_bin\futil.exe	- Intel PCI NIC FLASH programming utility. Will not update PXE in a system BIOS.
<temp>\pxe_bin\eeprom.exe	- Intel PCI NIC/LOM EEPROM programming utility.
<temp>\docs\	- Directory containing related documentation
<temp>\docs\pdkrel30.pdf	- Copy of the PXE Product Development Kit Instructions (what you are reading now)

After unpacking the files the install program will automatically run <temp>\disk1\setup.exe to install services and set up associated registry keys with default values.

When the install program completes, choose to reboot the system so the new services will be started correctly.

When you have installed the PDK, the following directories and files will exist below the install path:

<install directory>	-PDK installation directory
|---Uninst.dll	-Uninstall DLL
|---unpxe.inf	-Uninstall information
|	
+---DOCS	-Directory holding relevant documents
| |--pdkrel30.pdf	-Release document for this PDK
|	
+---SYSTEM	-Directory holding executables and boot files
| |---ApiTestOpts.dll	-API test DHCP option parser
| |---BStrapOpts.dll	-BSTRAP DHCP option parser
| |---PxeClientTester.dll	-PXE client test DHCP option parser
| |---DosUndiOpts.dll	-DOS/UNDI DHCP option parser
| |---PXEServerParser.dll	-PXE server DHCP option parser
| |---mkimage.exe	-Utility for creating bootfile file from floppy
| |---PxeConfig.exe	-Utility for configuring PXE settings
| |---PxeService.exe	-Bootserver and proxyDHCP Service
| |---PxeMTFTP.exe	-MTFTP Service
| |---PxeReg60.exe	-Utility to configure MS Dhcp service on Win2K
| |---PxeParser.dll	-PXE service option parser manager
| |---PXEmsg.dll	-PXE service messages
| \\---IMAGES	-Top of PXE image directory tree
| \\---X86PC	-Directory for X86PC specific boot files
| \\---UNDI	-Directory for UNDI specific boot files
| +---APITEST	-Directory for APITest Service boot files
| | |--apitest.0	-Layer 0 bootfile for APITest
| | |--mktest.bat	-Batch file for creating APITest.1 bootfile
| | \\---MKIMAGE	-Source files for APITest.1
| | |--autoexec.bat	-Source files for APITest.1
| | |--autoexec.tst	-Source files for APITest.1
| | |--config.sys	-Source files for APITest.1
| | |--copylogs.bat	-Source files for APITest.1
| | |--csagent2.exe	-Source files for APITest.1
| | |--keypluck.com	-Source files for APITest.1
| | |--keystuff.com	-Source files for APITest.1
| | |--ndistest.bat	-Source files for APITest.1
| | |--newtest.exe	-Source files for APITest.1
| | |--pxeapi.ini	-Source files for APITest.1
| | |--ramd.exe	-Source files for APITest.1
| | |--reboot.com	-Source files for APITest.1
| | \\---NET	-Source files for APITest.1
| | |--lmhosts	-Source files for APITest.1
| | |--ndis.dos	-Source files for APITest.1
| | |--netstart.bat	-Source files for APITest.1
| | |--protocol.ini	-Source files for APITest.1
| | |--system.ini	-Source files for APITest.1
| | \\--other files	-All other files installed from NT server CD
| |	
| +---bstrap	
| | bstrap.0	-Bstrap boot image
| +---RIStrap	
| | RIStrap.0	-Win 2000 RIS redirector image
| \\---DOSUNDI	-Directory for DOSUNDI Service boot files
| |--dosundi.0	-Layer 0 bootfile for DOSUNDI
| |--mkdos.bat	-Batch file for creating DOSUNDI.1 bootfile
| \\---MKIMAGE	-Source files for DOSUNDI.1
| |--autoexec.bat	-Source files for DOSUNDI.1
| |--config.sys	-Source files for DOSUNDI.1
| \\---NET	-Source files for DOSUNDI.1
| |--csagent2.exe	-Source files for DOSUNDI.1
| |--lmhosts	-Source files for DOSUNDI.1
| |--ndis.dos	-Source files for DOSUNDI.1
| |--netstart.bat	-Source files for DOSUNDI.1
| |--protocol.ini	-Source files for DOSUNDI.1
| |--system.ini	-Source files for DOSUNDI.1
| \\--other files	-All other files installed from NT server CD
|	
\\---TESTLOG	-Directory for test logs
|--file1	-Test files for NDIS testing
|--file2	-Test files for NDIS testing
|--file3	-Test files for NDIS testing

	--file4	-Test files for NDIS testing
	--file5	-Test files for NDIS testing
	--file6	-Test files for NDIS testing
	--file7	-Test files for NDIS testing
	--file8	-Test files for NDIS testing
	--file9	-Test files for NDIS testing
	--file10	-Test files for NDIS testing
	--file11	-Test files for NDIS testing
	--testfile.exe	-Executable to create test files FILE1-11

The installation will add a share to the "Testlog" directory. ***Please ensure that GUEST account has access rights to the "Testlog" directory and has permissions to connect to "Testlog" share.***

3.4.1 Create the boot files: APITEST.1 and DOSUNDI.1

Now create APITEST.1 and DOSUNDI.1 boot files (these are not provided with the PDK).
To create these files:

- Create two 1.44MB MSDOS 6.22 boot floppy diskettes using **FORMAT /S** or **FORMAT** and **SYS** from a DOS machine. You cannot do this from Windows NT or 2000.
- Label one diskette "APITest" and the other diskette "DOSUNDI".
- On the APITEST disk add the following DOS files in the root directory:

  HIMEM.SYS
  RAMDRIVE.SYS
  MORE.COM
  FC.EXE

To create the APITest.1 bootfile, insert the diskette labeled "APITest" in the drive and run the batch file:

<install directory>\ PDK\system\images\x86pc\undi\APITest\mktest.bat

To create the DOSUNDI.1 bootfile, insert the diskette labeled "DOSUNDI" and run the batch file:

<Install directory>\ PDK\system\images\x86pc\undi\DOSUNDI\mkdos.bat

(Shortcuts to both of these batch files can be found in the start menu.)

When the MKTEST.BAT and MKDOS.BAT files execute they will copy additional files to the diskette. Then a sector-by-sector image of the diskette is read into a file (either APITEST.1 or DOSUNDI.1) and the file is placed into the appropriate directory (either APITEST or DOSUNDI). This creates the boot files that are downloaded and booted by the client.

These two batch files will also verify the presence of network files needed by the dos client images to be generated. If you had skipped providing an NT 4.0 CD during installation, you can provide a location for the network files now by running *"mktest.bat –d <path>"* and/or *"mkdos.bat –d <path>".*

If you want to change the contents of the images, modify the files in the APITEST or DOSUNDI directories and run the batch files again. As an example, you may want to automatically log in to a server and run a test suite of your own by adding commands to the AUTOEXEC.BAT file in the DOSUNDI subdirectory.

Caution! File size, order of operation, resulting memory footprint, etc., are all critical to correct operation. Be aware of these restrictions and change image content with care, and only if necessary.

3.4.1.1 How to create other DOS boot files for your own testing

The utility <Install directory>\ PDK\system\MKIMAGE.EXE (used in the MKTEST.BAT and MKDOS.BAT batch files) is used to create a DOS boot diskette image (such as DOSUNDI.1 and APITEST.1). To use this utility, create a 1.44 MB bootable diskette (i.e.: Use DOS SYS.COM or FORMAT.COM /S to create a bootable diskette). You can leave the diskette as it is (containing only the MSDOS.SYS, IO.SYS, and COMMAND.COM files), or add AUTOEXEC.BAT and executables, etc. Whatever you put on the disk becomes part of the boot file image.

Place this diskette in drive A:\ of a computer and run MKIMAGE.EXE from a hard drive. This writes an image of the A:\ diskette into the current directory in a file named TEST.BIN (TEST.BIN will be 1.44 MB). TEST.BIN will need to be copied onto your boot server, into the appropriate image directory, and renamed to match the base name of your boot server type. For example, if you use the PXE Configuration Utility to create a new boot server type (say 123) and call it “MYDOS”, you will need to create this new image directory:

<Install directory>\ PDK\system\images\x86pc\undi\MYDOS

You will need a bootstrap environment layer image. This image is used to verify that the machine can run the OS image in the next layer, downloads the next layer and sets up the machine to properly run the next layer.

<Install directory>\ PDK\system\images\x86pc\undi\MYDOS\MYDOS.0

Finally, you will need to copy and rename your TEST.BIN to MYDOS.1.

<Install directory>\ PDK\system\images\x86pc\undi\MYDOS\MYDOS.1

3.5 Set up the PXE Client Workstation(s)

This PDK provides five binaries (E100_M.NIC, E100_S.NIC, E100_S.LOM, PXEBASE.NIC and PXEBASE.LOM) and a FLASH memory programming utility (FUTIL.EXE) to allow installing any of the binaries onto an Intel 82557/82558/82559 based network interface card (NIC).

E100_M.NIC is a monolithic (containing both PXE BaseCode and UNDI modules) implementation of a PXE-2.0 boot ROM. This image will only work when programmed into an Intel EtherExpress Pro PCI NIC FLASH device. This image will not work when loaded from the system BIOS FLASH device.

E100_S.NIC and E100_S.LOM are the UNDI modules in a split PXE-2.0 implementation (they do not contain a PXE BaseCode module). These boot ROMs will not work without a PXE BaseCode (PXEBASE.NIC or PXEBASE.LOM) module in the test system. PXEBASE.NIC can be programmed into an Intel NIC FLASH. PXEBASE.LOM must be included in the system BIOS image.

NOTE: The files ending in .NIC are designed to be programmed into FLASH memory devices in NICs. Do not use .NIC files in the system BIOS, they will not work.

NOTE: The files ending in .LOM are designed to be included in a system BIOS on the motherboard. They will operate correctly if programmed into a NIC. They use a lot more upper memory if the NIC is placed into a system that does not support POST Memory Manager.

Two NICs can be used to test split ROM implementations, program your UNDI module into one NIC and PXEBASE.NIC into the other. PXEBASE.NIC must be programmed into the FLASH memory on an Intel EtherExpress PRO PCI NIC. PXEBASE.LOM will not work if programmed into a PCI NIC FLASH memory device; this file will only work when included into a system BIOS that supports PXE split ROM images.

These binaries are provided as representative examples of “NIC-based” and “BIOS-based” PXE implementations to be used to create a known good test environment or for use in verifying correct operation of the BIOS in either a “LAN on Motherboard” platform, or a platform that uses a NIC.

NOTE: The PXEBASE.LOM ROM image is not a PCI device ROM image. It does not have a PCI Data Structure. The system BIOS must be modified to load this ROM image every time the system boots. If this ROM image is not loaded, PXE split ROMs will not be able to boot.

NOTE: Per this PDK’s license, you may not redistribute these binaries. They are provided for test purposes only.

3.5.1 Testing the PXE E100_S.LOM image

You can incorporate the E100_S.LOM into your client machine two ways.

1. The first way is to program the image into the FLASH memory on an Intel EtherExpress PRO PCI NIC using futil.exe:

   futil E100_S.LOM

2. The second way is to integrate the E100_S.LOM into the BIOS of platforms that contain the Intel EtherExpress Pro PCI network device on the motherboard. The E100_S.LOM image must be installed as an option ROM image in your system BIOS image (as you would include a video option ROM image). Use of this binary requires building the BIOS with this binary and then updating the BIOS of the system under test. You must use your own BIOS programming utility to program this BIOS image into the motherboard's FLASH memory. This operation will normally only be done by the BIOS engineering department of the motherboard manufacturer.

For the PXE LOM image to boot, a separate PXE BaseCode image must be available in the system. This can be accomplished in two ways:

1. Program PXEBASE.NIC into the FLASH memory on an Intel NIC using futil.exe:

   futil PXEBASE.NIC

2. Include PXEBASE.LOM in the system BIOS. This ROM image should always be loaded by the system BIOS.

3.5.1.1 FUTIL.EXE Usage

The FLASH utility, futil.exe, will only program and erase FLASH parts on an Intel EtherExpress PRO NIC. This utility will not program PXE ROM images into the system BIOS FLASH device(s).

(The following is also displayed by typing: FUTIL /?)

```
FUTIL v3.21 - Intel PCI NIC FLASH Update Utility
Copyright (C), 1995 - 1999  Intel Corporation

Usage: FUTIL command/filename

Commands:
      -help
           Brings up command line help.
      -bootdisable or -bd
           Disables Boot ROM.
      -bootenable or -be
           Enables Boot ROM.
      -dump
           Dumps FLASH memory contents to file.
      -nic=
           Selects a specific adapter (1-8).
      -erase
           Erases the contents of the FLASH memory.
      -verify <filename>
           Verifies the contents of the FLASH memory.
           filename.
      -Object imagefile objectfile
           Create a HexaDecimal Object Code 88 from
           the imagefile.
      filename
           Programs a raw binary image into FLASH memory.

      NOTE: To view all adapters in system, run Futil
            without any command line options.
Exit codes:
     0 := All FLASH operations completed successfully.
     1 := Bad command line parameter.
     2 := No supported supported PCI network adapters detected.
     3 := No supported FLASH devices detected.
     4 := FLASH operation failed
     5 := Image file is missing or corrupted.
```

3.5.1.2 More FUTIL.EXE Information

```
Intel PCI NIC FLASH Update Utility
Version 3.21
=================================================================

DISCLAIMER:
-----------
Copyright (c) 1998-1999, Intel Corporation. All rights reserved. Intel
```

```
Corporation, 5200 NE Elam Young Parkway, Hillsboro, OR 97124-6497.
Intel Corporation assumes no responsibility for errors or omissions in
this document or software. Nor does Intel make any commitment to update
the information or software contained herein. Software product is
supplied "as is" and may not be fully functional.

Other product and corporate names may be trademarks of other companies
and are used only for explanation and to the owners' benefit, without
intent to infringe.


RUNNING FUTIL
------------

FUTIL.EXE can be run with any of the following command line options:


OPTIONS:
--------

   -? or -HELP     - Brings up command line help.
   -BOOTENABLE         - Enables Boot ROM.
   -BOOTDISABLE        - Disables Boot ROM.
   -DUMP           - Dumps flash memory contents to file.
   -ERASE          - Erases Flash memory.
   -NIC=           - Selects a specific adapter (1-8).
   -VERIFY <file>  - Verify Flash memory against <file>.

EXAMPLES:
--------
Example 1:
To program the first Intel Network adapter
in your system, call FUTIL.EXE like this:
   FUTIL -nic=1 <filename>

Example 2:
To program *all* of the Intel Network adapters
in your system , call FUTIL.EXE like this:
   FUTIL <filename>

Example 3:
To verify the Flash Memory on the second Intel
Network adapter in your system against a raw
flash image file, call FUTIL.EXE like this:
   FUTIL -nic=2 -verify <filename>

Example 4:
To dump all of the current flash memory contents
to a file, call FUTIL.EXE like this:
   FUTIL -dump

NOTE:
-----

* To view all adapters in system, run FUTIL
  without any command line options.

* FUTIL will attempt to erase/program all Intel
  Network adapters that it finds, unless you specify
  which adapters you want to erase/program, via the
  "-nic=" option.

* When using the '-dump' command, FUTIL will automatically
  create a file and name it, based on the last 6 bytes
  of your Intel Network adapter's MAC Address.  For example,
  if your MAC Address was '00AA11223344', FUTIL would
  create the file called '11223344.FLS'.
```

```
DOS ERROR CODES:
----------------
FUTIL returns error codes to the DOS command line; a
description of the error codes is found below.

CODE NAME:               VALUE:
----------               ------
SUCCESSFUL                  0
BAD CMD LINE PARAMETER      1
NO PCI NIC FOUND            2
NO FLASH                    3
BAD FLASH                   4
BAD FILE                    5


SUPPORTED FLASH DEVICES:
------------------------

VENDOR:     PART#:
-------     ---------
AMD         28F020
ATMEL       AT49BV512
ATMEL       AT29LV512
ATMEL       AT49LV010
CATALYST    28F512
CATALYST    28F010
CATALYST    28F020
INTEL       28F010
INTEL       28F020
SGS         28F512
SST         29LE512
SST         39SF512
SST         39VF512
ST MICRO    29W512B
```

3.5.2 Boot ROM Setup and Operation

This section only applies to Intel NICs with PXE boot ROMs made using the images (E100_M.NIC and E100_S.NIC) included in this PDK.

The first time PXE is programmed into a NIC, it will default to BBS for booting. Because some older systems do not support booting option ROMs via BBS a DOS utility, eeprom.exe, has been included in this PDK that can be used to change the PXE bootstrap mechanism.

PXE supports four bootstrap selections: BBS, interrupt 18h, interrupt 19h and disabled.

To boot using BBS do this: eeprom -bbs

To boot using interrupt 18h do this: eeprom -18h

To boot using interrupt 19h do this: eeprom -19h

To disable network booting do this: eeprom -disablenet

In addition to selecting the bootstrap, eeprom.exe can be used to tell monolithic PXE (E100_M.NIC) to use the BaseCode that is included in the system BIOS instead of using the BaseCode that is included in the NIC FLASH. If there is no BaseCode in the system BIOS, the BaseCode in the NIC FLASH will be used.

To use the BaseCode in the system BIOS (if present) do this: eeprom -enablebc

To only use the BaseCode in the NIC FLASH do this: eeprom -disablebc

3.5.2.1 Enabling Network Boot

After installing PXE onto the NIC or motherboard, make the NIC the first boot device.

The ability to make the NIC the first boot device depends on underlying BIOS support. WfM compliance requires BIOS Boot Spec (BBS) support. If BBS suport is present, setting the boot order should not be a problem. To set the boot order in a client for most BBS-compliant systems:

- Enter BIOS setup during POST
- Locate the boot order setting
- Make the NIC the first boot device

(Even if your platform does not support BBS, you may still be able to reorder the NIC to the first boot device in the BIOS setup as many manufacturers have implemented proprietary methods.)

3.5.2.1.1 Troubleshooting Boot Order Selection Problems

The Intel PXE boot ROM may not work properly if the bootstrap interrupt is set incorrectly. Possible problems and solutions are:

1. PXE will not boot or only boots when there is no other bootable device in the system. This can happen if interrupt 18h bootstrap is used in an older platform.

   To correct this problem, reboot using a DOS diskette and select interrupt 19h using: eeprom.exe -19h

2. PXE always boots first, even though 'Network' is not the first item in the BIOS startup list. This can happen if interrupt 19h is used in a system that supports Interrupt 18h boot device reordering or BBS.

   To correct this problem, reboot reboot using a DOS diskette and select interrupt 18h using: eeprom.exe -18h

3.5.2.2 Canceling a Network Boot

<Esc> or **<Ctrl+C>** ***Cancel network boot***

Once the NIC with PXE has been selected as the network boot device, the user can cancel at any time during network boot by pressing <Esc> or <Ctrl+C>. When one of these keys is pressed, PXE will reset the NIC, remove itself from RAM, and return control to the BIOS.

3.6 PXE Client Status/Error Codes

3.6.1 Init/Boot/Loader Codes

3.6.1.1 PXE-E00: Could not find enough free base memory.

PXE BaseCode and UNDI runtime modules are copied from FLASH or upper memory into the top of free base memory between 480K (78000h) and 640K (A0000h). This memory must be zero filled by the system BIOS. If this memory is not zero filled, the relocation code in the PXE ROMs will assume that this memory is being used by the system BIOS or other boot ROMs.

3.6.1.2 PXE-E01: PCI Vendor and Device IDs do not match!

This message should never be seen in a production BIOS. When the system BIOS initializes a PCI option ROM, it is supposed to pass the PCI bus/device/function numbers in the AX register. If the PCI device defined in the AX register does not match the UNDI device, this error is displayed.

3.6.1.3 PXE-E04: Error reading PCI configuration space.

This message is displayed if any of the PCI BIOS calls made to read the PCI configuration space return an error code. This should not happen with a production BIOS and properly operating hardware.

3.6.1.4 PXE-E05: EEPROM checksum error.

This message is displayed if the NIC EEPROM contents have been corrupted. This can happen if the system is reset or powered down when the NIC EEPROM is being reprogrammed. If this message is displayed the PXE ROM will not boot.

3.6.1.5 PXE-E06: Option ROM requires DDIM support.

This message should not be seen in a production BIOS. PCI option ROMs must always be installed as DDIM option ROMs (they must be installed into read/write upper memory).

3.6.1.6 PXE-E07: PCI BIOS calls not supported.

This message should not be seen in a production BIOS. PCI BIOS must have PCI BIOS services.

3.6.1.7 PXE-E08: Unexpected API error. API: xxxxh Status: xxxxh

This message is displayed if a PXE API returns a status code that is not expected by the runtime loader.

3.6.1.8 PXE-E09: Unexpected UNDI loader error. Status: xxxxh

This message is displayed if the UNDI runtime loader returns an unexpected status code.

3.6.2 ARP Codes

3.6.2.1 PXE-E11: ARP timeout.

The PXE ROM will retry the ARP request four times, if it does not get any valid ARP replies, this message is displayed. This error can be caused by a number of network and service configuration errors. The most common are:

- Setting the DHCP Class Identifier (option 60) on the DHCP server and installing the proxyDHCP on a separate machine.
- Using routers that do not respond to ARP requests.

3.6.3 BIOS and BIS Codes

3.6.3.1 PXE-E20: BIOS extended memory copy error. AH == nn

This message is displayed if the BIOS extended memory copy service returns an error. This should not happen on a production BIOS. ***nn*** is the BIOS error code returned by the BIOS extended memory copy service (Int 15h, AH = 87h).

3.6.3.2 PXE-E21: BIS integrity check failed.

This message is displayed if the BIS image in extended memory has been corrupted.

3.6.3.3 PXE-E22: BIS image/credential validation failed.

The downloaded image and credential do not match the client key.

3.6.3.4 PXE-E23: BIS initialization failed.

BIS could not be initialized. No more data is available.

3.6.3.5 PXE-E24: BIS shutdown failed.

BIS could not be shutdown. No more data is available.

3.6.3.6 PXE-E25: BIS get boot object authorization check flag failed.

Could not determine if BIS is enabled/disabled.

3.6.3.7 PXE-E26: BIS free memory failed.

Could not release BIS allocated memory.

3.6.3.8 PXE-E27: BIS get signature information failed.

Required BIS credential type information could not be determined.

3.6.3.9 PXE-E28: BIS bad entry structure checksum.

BIS entry structure in the SM BIOS table is invalid.

3.6.4 TFTP/MTFTP Codes

3.6.4.1 PXE-E32: TFTP open timeout.

TFTP open request was not acknowledged. Verify that the TFTP service is running.

3.6.4.2 PXE-E35: TFTP read timeout.

Next TFTP data packet was not received.

3.6.4.3 PXE-E36: Error received from TFTP server.

A TFTP error packet was received from the TFTP server.

3.6.4.4 PXE-E38: TFTP cannot open connection.

A hardware error occurred when trying to send the TFTP open packet out.

3.6.4.5 PXE-E39: TFTP cannot read from connection.

A hardware error occurred when trying to send a TFTP acknowledge packet out.

3.6.4.6 PXE-E3A: TFTP too many packages.

This message can mean one of two things. 1 – You are trying to download a file using TFTP that is larger than the allocated buffer. 2 – You started downloading a file as a slave client using MTFTP and the file increased in size when you became the master client.

3.6.4.7 PXE-E3B: TFTP error – File not found.

The requested file was not found on the TFTP server.

3.6.4.8 PXE-E3C: TFTP error – Access violation.

The request file was found on the TFTP server. The TFTP service does not have enough access rights to open/read the file.

3.6.4.9 PXE-E3F: TFTP packet size is invalid.

The TFTP packet received is larger than 1456 bytes.

3.6.5 BOOTP/DHCP Codes

3.6.5.1 PXE-E51: No DHCP or proxyDHCP offers were received.

The client did not receive any valid DHCP, BOOTP or proxyDHCP offers.

3.6.5.2 PXE-E52: proxyDHCP offers were received. No DHCP offers were received..

The client did not receive any valid DHCP or BOOTP offers. The client did receive at least one valid proxyDHCP offer.

3.6.5.3 PXE-E53: No boot filename received.

The client received at least one valid DHCP/BOOTP offer, but does not have a boot filename to download.

3.6.5.4 PXE-E55: proxyDHCP service did not reply to request on port 4011.

The client issued a proxyDHCP request to the DHCP server on port 4011 and did not receive a reply.

3.6.6 UNDI Codes

3.6.6.1 PXE-E60: Invalid UNDI API function number.

An API being used by the BaseCode is not implemented in the UNDI ROM.

3.6.6.2 PXE-E61: Media test failed, check cable.

Most likely the cable is not plugged in or connected. Could be a bad cable, NIC or connection.

3.6.6.3 PXE-E63: Error while initializing the NIC.

An error occurred while trying to initialize the NIC hardware. Try another NIC.

3.6.6.4 PXE-E64: Error while initializing the PHY.

An error occurred while trying to initialize the PHY hardware. Try another NIC.

3.6.6.5 PXE-E65: Error while reading the configuration data.

An error occurred while reading the NIC configuration data. Try another NIC.

3.6.6.6 PXE-E66: Error while reading the initialization data.

An error occurred while reading the NIC initialization data. Try another NIC.

3.6.6.7 PXE-E67: Invalid MAC address.

The MAC address stored in this NIC is invalid. Try another NIC.

3.6.6.8 PXE-E68: Invalid EEPROM checksum.

The EEPROM checksum is invalid. The contents of the EEPROM have been corrupted. Try another NIC.

3.6.6.9 PXE-E69: Error while setting interrupt.

The interrupt hardware could not be configured. Try another NIC.

3.6.7 Bootstrap and Discovery Codes

3.6.7.1 PXE-E74: Bad or missing PXE menu and/or prompt information.

PXE tags were detected but the boot menu and/or boot prompt tags were not found/valid.

3.6.7.2 PXE-E76: Bad or missing multicast discovery address.

Multicast discovery is enabled but the multicast discovery address tag is missing.

3.6.7.3 PXE-E77: Bad or missing discovery server list.

Multicast and broadcast discovery are both disabled, or use server list is enabled, and the server list tag was not found/valid.

3.6.7.4 PXE-E78: Could not locate boot server.

A valid boot server reply was not received by the client.

3.6.7.5 PXE-E79: NBP is too big to fit in free base memory.

The NBP is larger than the amount of free base memory.

3.6.7.6 PXE-E7A: Client could not locate a secure server.

This message is displayed when the client did not receive any security information from the boot server and BIS is enabled on the client.

3.6.7.7 PXE-E7B: Missing MTFTP server IP address.

This message is displayed when the ROM did not receive any PXE discovery tags or proxyDHCP offers and the DHCP SIADDR field is set to 0.0.0.0.

3.6.8 Miscellaneous Codes

3.6.8.1 PXE-EA0: Network boot canceled by keystroke.

User pressed <Esc> or <Ctrl-C> during DHCP/Discovery/TFTP.

3.6.9 BaseCode/UNDI Loader Codes

3.6.9.1 PXE-EC1: BaseCode ROM ID structure was not found.

UNDI boot module could not find the BaseCode ROM ID structure. If there is a BaseCode ROM image in the system, it has probably been corrupted.

3.6.9.2 PXE-EC3: BaseCode ROM ID structure is invalid.

The BaseCode ROM ID structure is invalid. The BaseCode ROM image has probably been corrupted.

3.6.9.3 PXE-EC4: UNDI ROM ID structure was not found.

The BaseCode loader module could not locate the UNDI ROM ID structure.

3.6.9.4 PXE-EC5: UNDI ROM ID structure is invalid.

The UNDI ROM image has probably been corrupted.

3.6.9.5 PXE-EC6: UNDI driver image is invalid.

The UNDI ROM image has probably been corrupted.

3.6.9.6 PXE-EC8: !PXE structure was not found in UNDI driver code segment.

The UNDI ROM image has probably been corrupted, or has not been initialized by the BIOS. This error is most often caused by one of three things:

- A .NIC image was programmed into a BIOS when a .LOM image should have been used.
- The memory allocated by the POST Memory Manager ($PMM) during PXE option ROM initialization has been corrupted or erased before PXE option ROM boot.
- The UNDI_Loader structure was not properly initialized during option ROM initialization.

3.6.9.7 PXE-EC9: PXENV+ structure was not found in UNDI driver code segment.

The UNDI ROM image has probably been corrupted, or has not been initialized by the BIOS. This error is most often caused by one of three things:

- A .NIC image was programmed into a BIOS when a .LOM image should have been used.
- The memory allocated by the POST Memory Manager ($PMM) during PXE option ROM initialization has been corrupted or erased before PXE option ROM boot.
- The UNDI_Loader structure was not properly initialized during option ROM initialization.

4. PXE Services Configuration

4.1 Overview

PxeService provides the following capabilities:

- PxeService in *"proxyDHCP server mode"* supplements the DHCP service by providing DHCP options that cannot be programmed, or that you prefer not to program, into the DHCP service. The options provided by the service in this mode provide the client with a list of available Bootserver types and other information that allows the client to automatically construct a boot menu.
    - proxyDHCP is also used to provide a special bootfile name (bstrap.0) to pre-version 2.0 PXE clients. This bootfile is used by the pre PXE 2.0 client to do Bootserver discovery. (This capability is built into version 2.0 PXE clients.)
- PxeService in *"Bootserver mode"* responds to the "discovery request" from the booting client and provides configuration specifics to the client type and the bootfile name.

MTFTP service provides

- Multicast and unicast TFTP service for downloading the bootfile whose name was provided by PxeService (in either proxyDHCP or Bootserver mode). MTFTPD may also be used by the executing bootfile to download subsequent boot files.

The PXE Configuration Utility allows

- Setting up and subsequent modification of PxeService and MTFTPD service.
- Configuring multiple servers from one location.

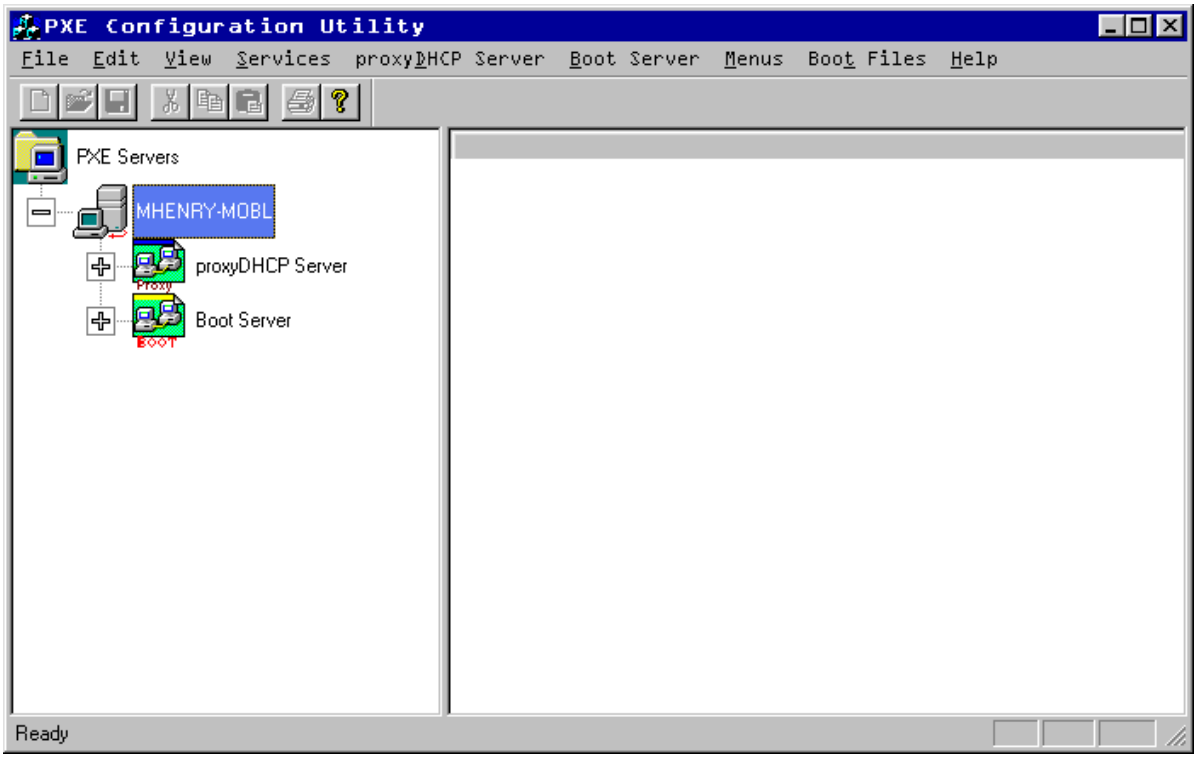
  For example, you may choose to run PxeService in proxyDHCP mode on the DHCP server and PxeService in Bootserver mode on one or more additional servers. These servers can all have their PXE and MTFTD services configured from a single location.

4.2 Starting the PXE Configuration Utility

Before using the PXE Configuration Utility:

- Verify that the PXE PDK has been installed and "setup" run on all machines to be configured.
- Run <install directory>\system\PxeConfig.exe on the machine from which you intend to configure all machines. (Note: the version of this capability has limitations. You must have an identical account and password on all the machines to be configured as the machine you are using to run PXE Configuration Utility.)

The PDK installation has already done basic configuration of the services. As a result, the PXE Configuration Utility startup window will look similar to the screen shot below.



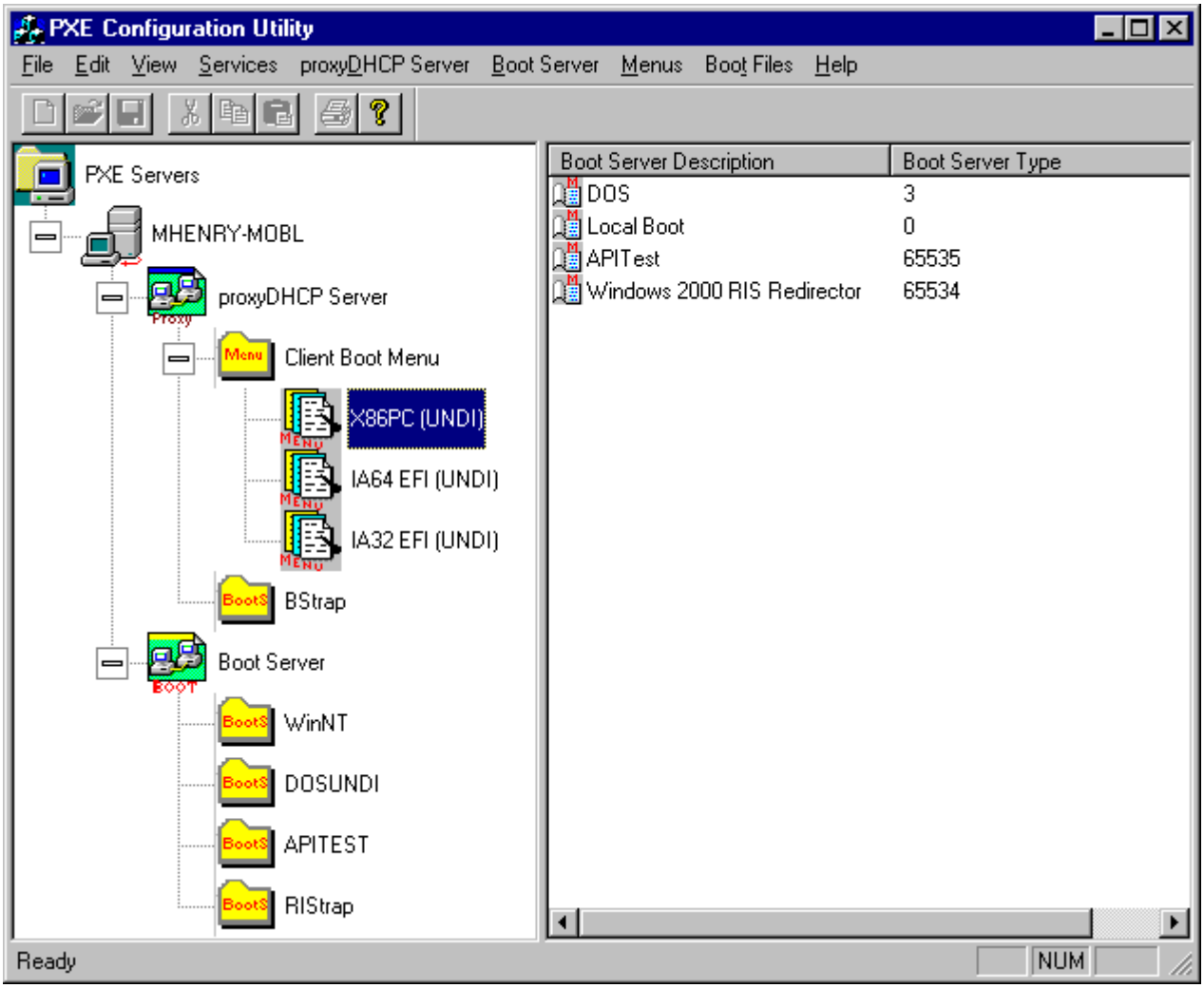
Expanding the items in the left windowpane (see figure below), we find that the proxyDHCP server is configured to support boot clients of the X86PC architecture that have a network interface with an UNDI API. It is also configured for IA32 and IA64 EFI platforms.

The bootservers installed by the PDK setup are DOSUNDI, APITEST and RIStrap.

DOSUNDI supports booting of the client to DOS platform by the dosundi.0 and dosundi.1 images.

APITEST provides images apitest.0 and apitest.1 to the client for testing UNDI APIs, ROM structure, network packet compatibility and NDIS driver working.

RIStrap is bootservice that redirects a client to an MS Windows 2000 Remote Installation Server (RIS) on the network. RIStrap provides functionality of bootserver arbitration for RIS since RIS does not support it yet. For RIStrap to operate correctly, it is recommended to have RIS server on a machine other than the DHCP or PXE server.

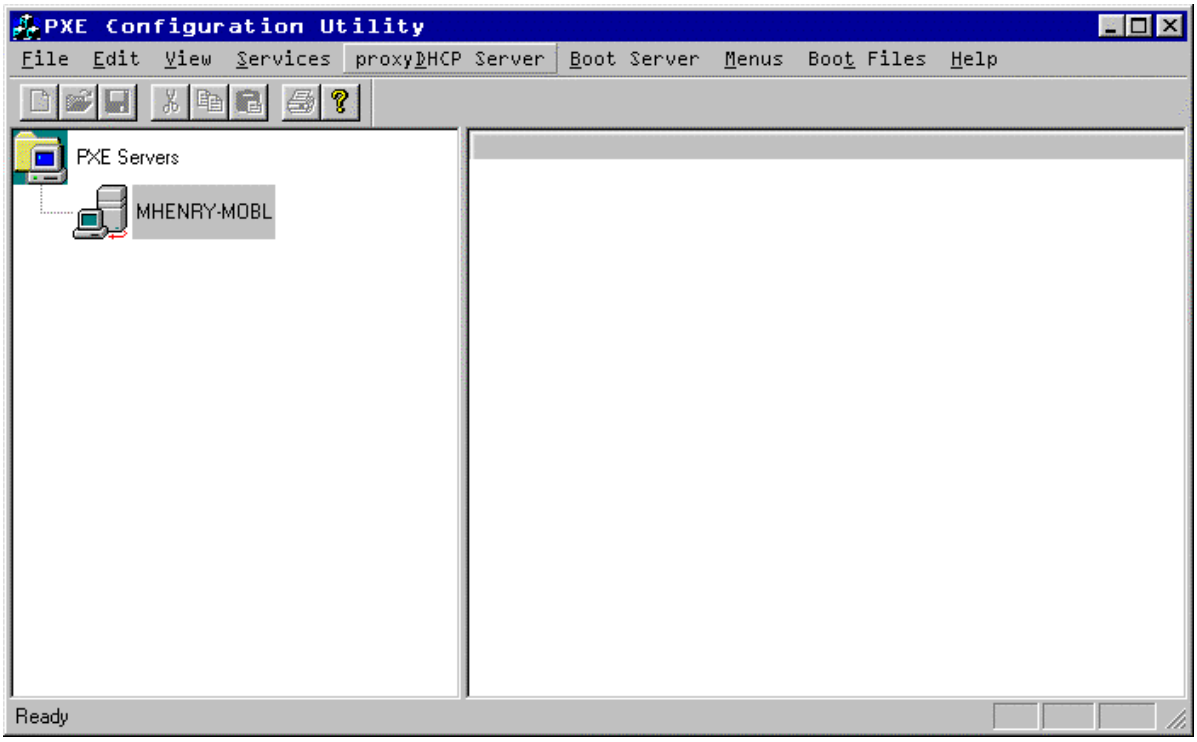


Booting clients will be supplied with a menu of Bootserver types. These Bootservers (Local Boot, DOSUNDI, APITEST and Windows 2000 RIS Redirector) are shown in the right pane of the window. The default boot server (that is, the one that the client will use if the user does not intervene) in this example is “Local Boot”. “Local Boot” is a special case. In effect, when indicating “Local Boot” as a Bootserver, proxyDHCP is telling the booting client to silently fail the network boot and drop to the next boot device in the BIOS boot order. (For example, “A:” or “C:”)

4.3 Using the PXE Configuration Utility

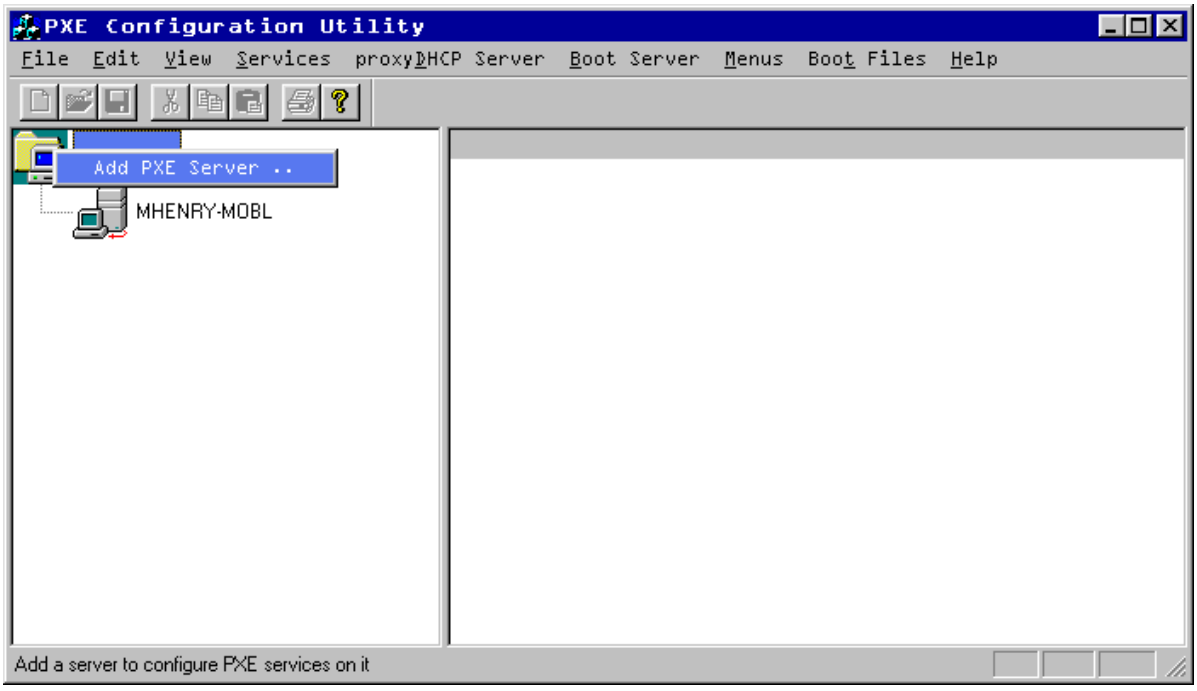
As an example, this section describes configuring a single PXE server with the same default configuration provided by the PDK install program. We start with a test platform where:

- The PDK has been installed.
- Both PxeService and MTFTPD service are installed but stopped.
- PxeService configuration includes neither proxyDHCP nor Bootserver servers.

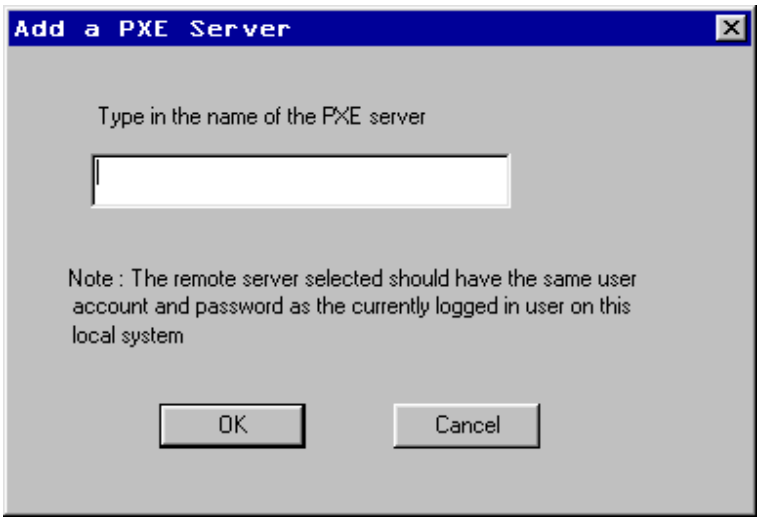


4.3.1 Select PXE Servers

To select more PXE servers for configuration, right-click the PXE servers icon and then left-click on Add PXE Server.



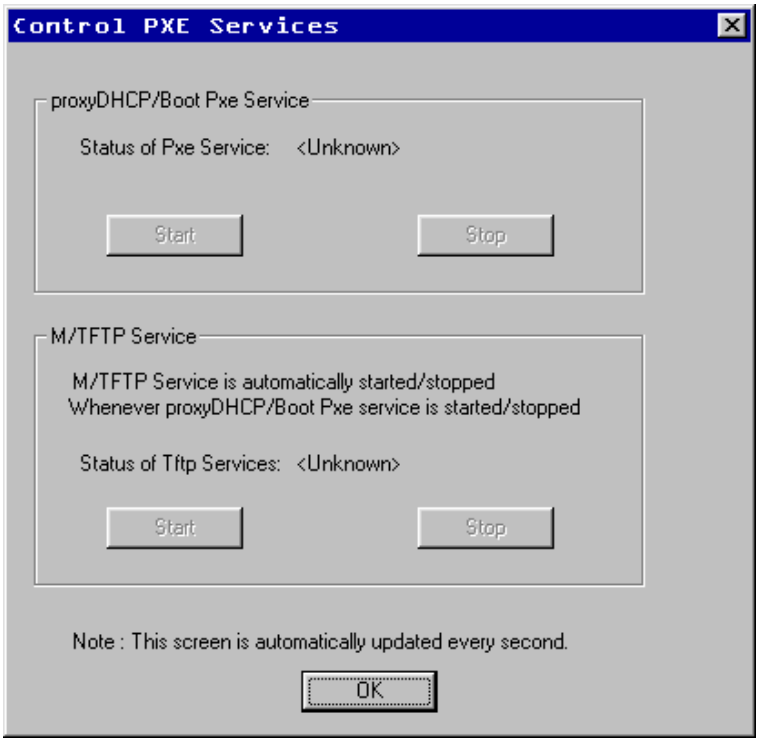
This will bring up the following dialog. Note: The remote server selected must have the same user account and password as the currently logged-in user on this local machine.



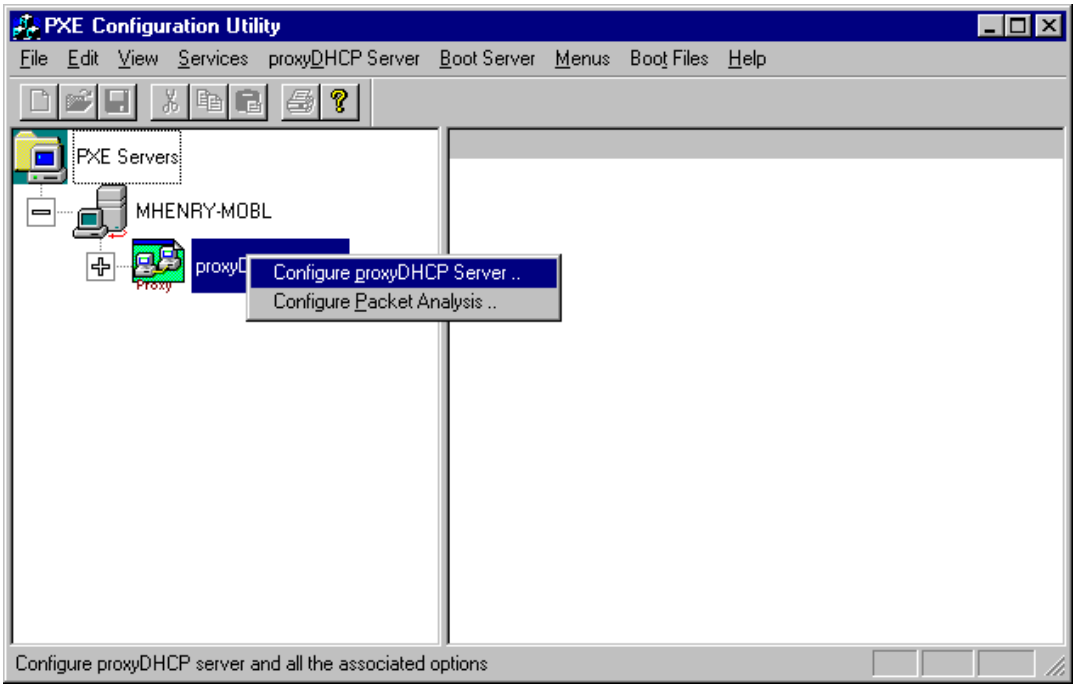
4.3.2 Configure proxyDHCP

To enable proxyDHCP, right-click on the particular PXE server being configured. Select Enable proxyDHCP Server. Note that this pull-down menu also provides a PXE services Start/Stop toggle. You can use this feature to start and stop PXE services on remote machines.

The Start/Stop… selection brings up a dialog that allows starting and stopping the PxeService and MTFTPD services. (Again, both locally and remotely.)



To configure proxyDHCP, right-click the proxyDHCP icon.



Selecting Configure proxyDHCP Server … brings up the dialog box in the next section. The tabs define configuration items specific to the proxyDHCP service, information provided by proxyDHCP to the booting client, and, similarly, information provided to inquiring Bootservers.

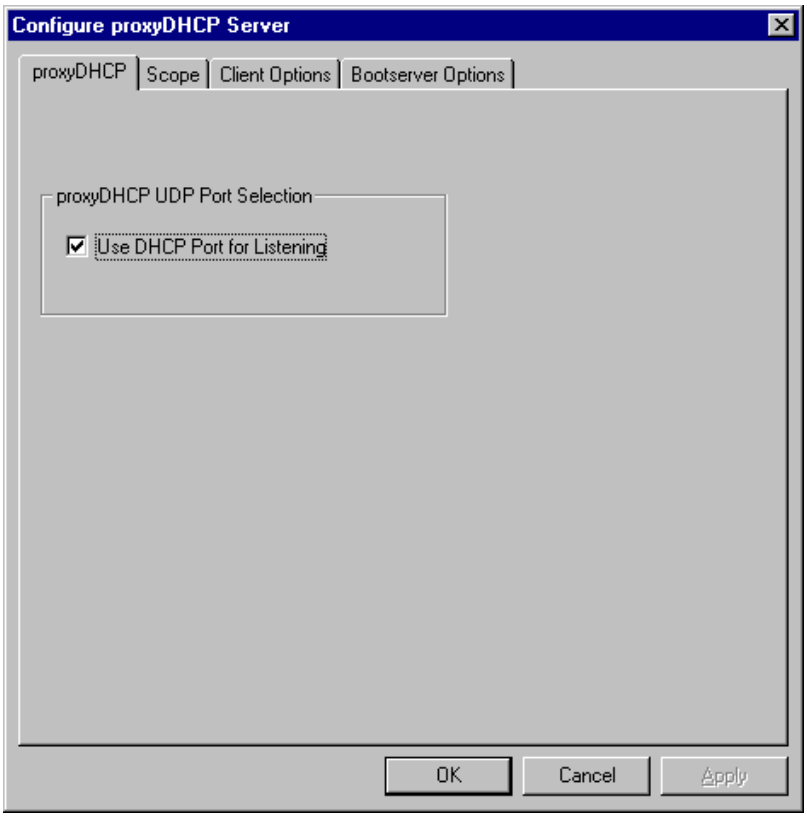
4.3.2.1 proxyDHCP UDP Port Selection

The listening UDP ports used by proxyDHCP are controlled by this checkbox. (The checkbox programs registry flag **UseDHCPPort**. If you are viewing this document with Microsoft Word for Office97, the registry flag name is a hyperlink to the respective entry in the Registry Entries section below.)

- If this registry flag is set to 1, proxyDHCP will open a UDP socket on port 67 and 4011.
- If this registry flag is set to 0, proxyDHCP will open a UDP socket on port 4011 only.

If the proxyDHCP service is running on the same host as the DHCP service, UseDHCPPort must be set to 0 because port 67 is used by the DHCP service. In this case, proxyDHCP will only process request packets sent to port 4011.

The following Dialog box defines all of the above:

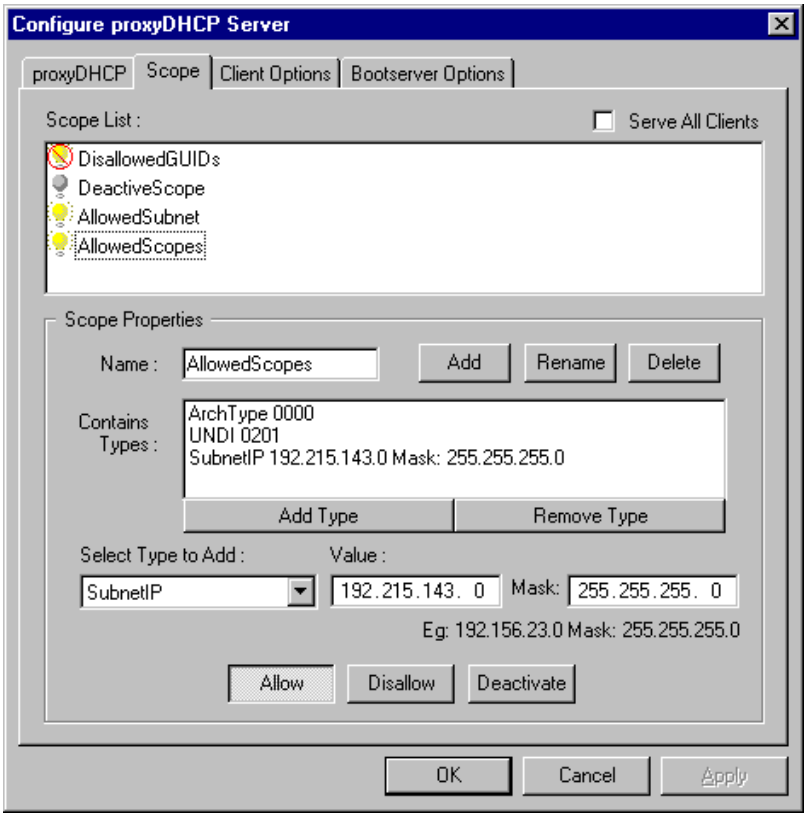


4.3.2.2 Scoping

The proxyDHCP server can filter the clients to which it will respond. Using the scope feature, the server can be made to handle only selected types of clients and reply to them appropriately.

The clients can be scoped on the basis of the following 5 criteria:

Subnet IP	: clients from particular subnets
Class ID (option 60)	: clients responding with definite Class ID value.
GUID (option 97)	: Individual client identification UUID number
UNDI version (option 94)	: some version of UNDI
Architecture type (option 93)	: whether the client is Intel PC, NEC or other.



The user is allowed to create scopes that identify one or more client machines. Multiple scopes can be Allowed / Disallowed / Deactivated to get a combined effect.

To create a new scope, type a name in the Name field and click Add. The new scope appears in the Scope List.

You can then add contents to the scope. Select Type to Add from the list, then provide an appropriate Value. Click Add Type. Additional information about the value is displayed on a line below. The new content appears in the Contains list. Additional Types can be added using this method.

The Allow / Disallow / Deactivate buttons provide enabling for inclusion, enabling for exclusion and disabling for dormancy of a scope respectively. In case of a client falling in multiple scopes, Exclusion (Disallow) will have higher precedence than Inclusion (Allow). The PXE server will not respond to a client in a disallowed scope. It will only respond to a client in an allowed scope. The exclusion scoping makes it easy to avoid a small group of clients from a larger group.

The Serve All Clients check box bypasses all scopes, so that PXE Server responds to all clients.

A scope name can be changed by selecting that particular scope, typing the new name in Name field and clicking Rename.

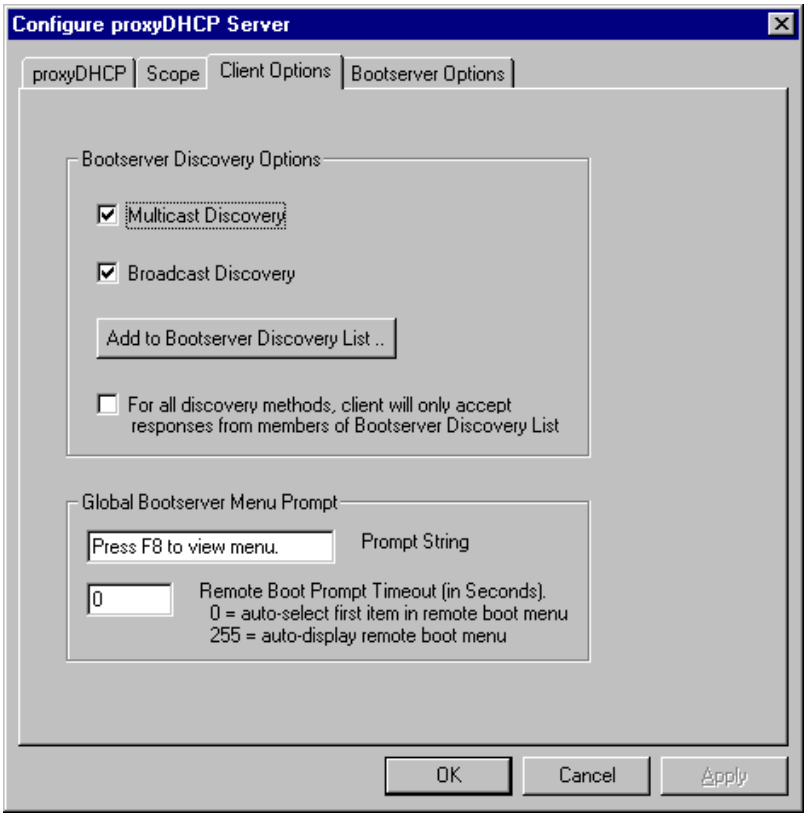
A scope can be deleted using the Delete button.

4.3.2.3 Global proxyDHCP Configuration Information

4.3.2.3.1 proxyDHCP to Client Configuration Information

Some configuration information supplied to the booting client by proxyDHCP is specific to the client's system architecture, and some information is global to all requesting clients, regardless of system architecture.

Global boot client configuration information is defined in this tab.



4.1.3.2.3.1 Bootserver Discovery Methods

The booting client discovers Bootservers by type. The client can look for a Bootserver using broadcast, multicast or unicast messages. This tab allows the administrator to define what type(s) of discovery method(s) will be performed by the client(s). If no discovery method is provided, the client will abort the remote boot.

The Bootserver discovery methods are listed in order of precedence. For example, if all three methods are enabled, the booting client will attempt to locate its Bootserver using Multicast Discovery. If this fails, the client will then try broadcasting for a Bootserver, and finally, if all else fails, the client will attempt to find its Bootserver unicasting to the explicit list of Bootservers in the Bootserver Discovery List.

The various discovery methods have advantages and disadvantages as discussed in the following sections.

4.3.2.3.1.1.1 Multicast

This method is useful when intervening routers between the client and Bootserver have been configured to pass multicast messages.

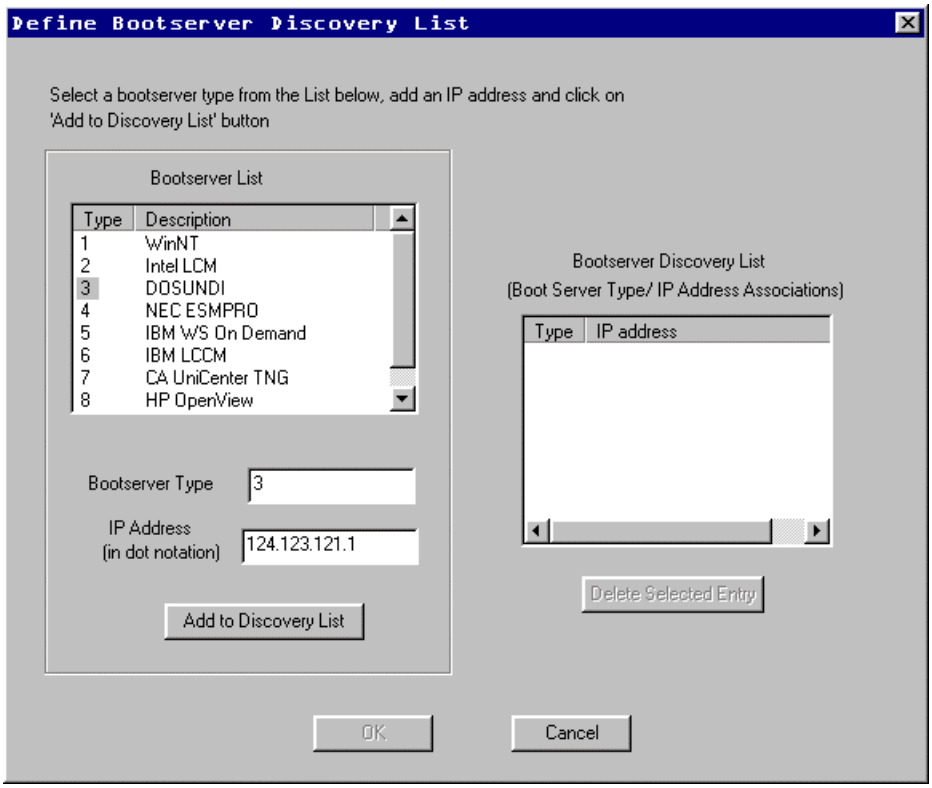
4.3.2.3.1.1.2 Broadcast

The client sends a broadcast Bootserver Discover message to UDP port 67. This method is only useful if the Bootserver is on the same segment as the booting client, or if the router BOOTP relay agents have been configured to pass broadcast messages to particular Bootservers. Also, this method will not work if the Bootserver is on the same server as the DHCP service. All client broadcasts will be to UDP port 67 and it is not possible for Bootserver and DHCP to share the same UDP port.

4.3.2.3.1.1.3 Unicast

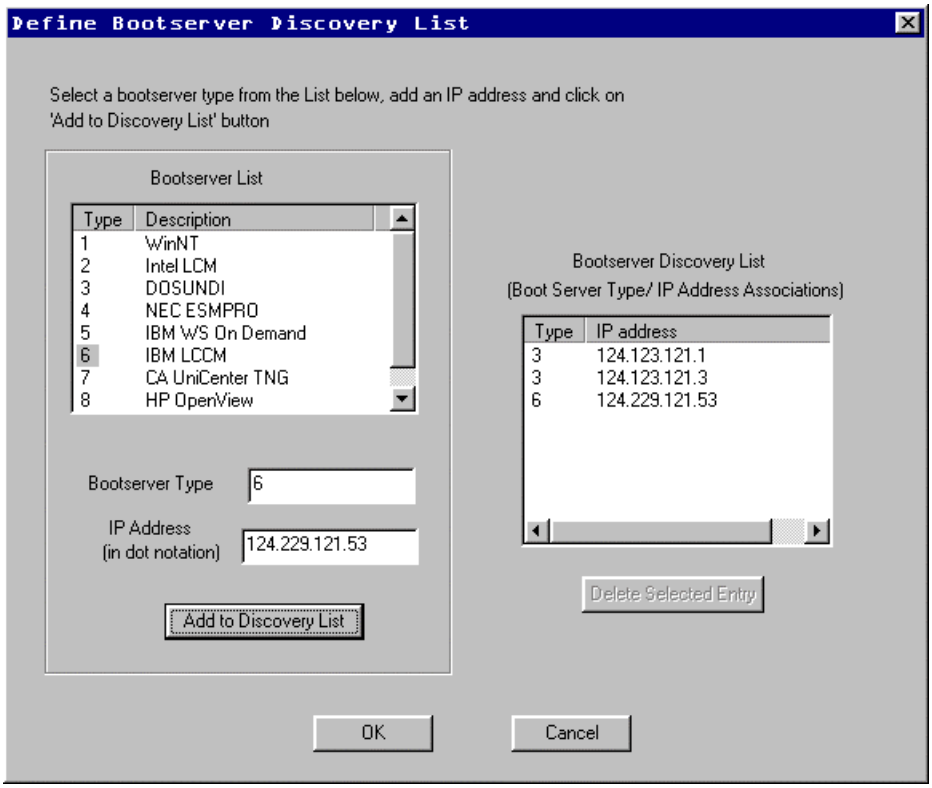
The proxyDHCP service may also provide the client with an explicit list of IP addresses for each Bootserver in the boot menu. In this case the client will contact the Bootserver directly. No broadcast or multicast messages are involved.

The Bootserver List is defined in a dialog box presented by selecting the Add to Bootserver Discovery List button.



The list is formed by selecting a bootserver type and associating specific IP addresses, one at a time with the type.

For example, the following figure defines a Bootserver List comprised of one IBM LCCM bootserver and two DOSUNDI test Bootservers.



4.3.2.3.1.1.4 Second Use for the Unicast List

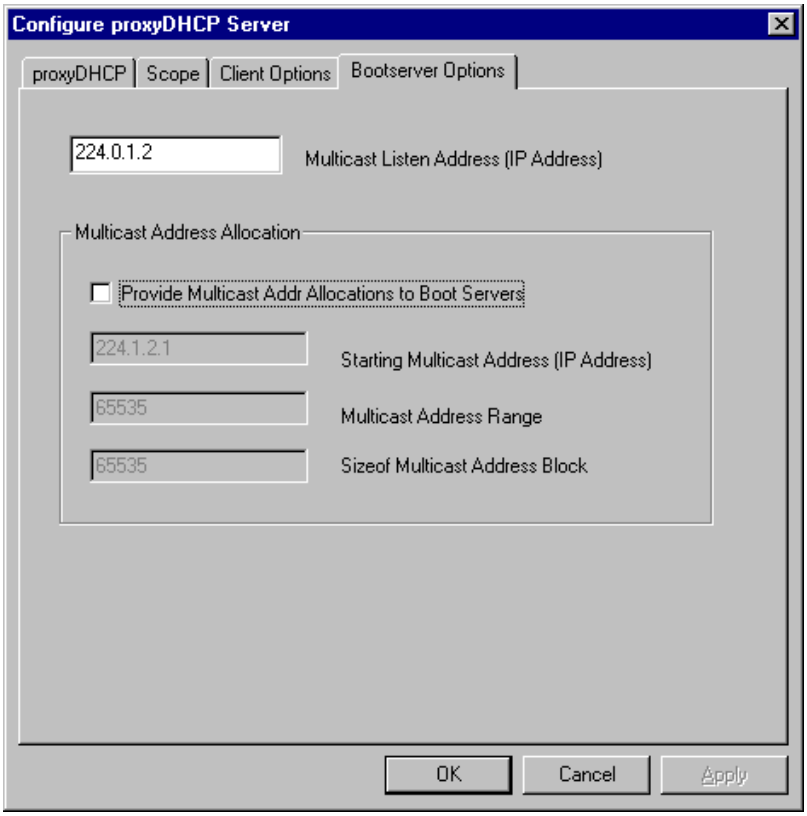
In addition, the unicast list may be used to qualify the use of Bootserver responses received via either broadcast or multicast discovery. In this case, the client will check the IP address of the responding server against the list of IP addresses provided for that server type. This mode of operation is enabled by checking the Server List Only check box.

4.3.2.3.1.1.5 Global Bootserver Menu Prompt

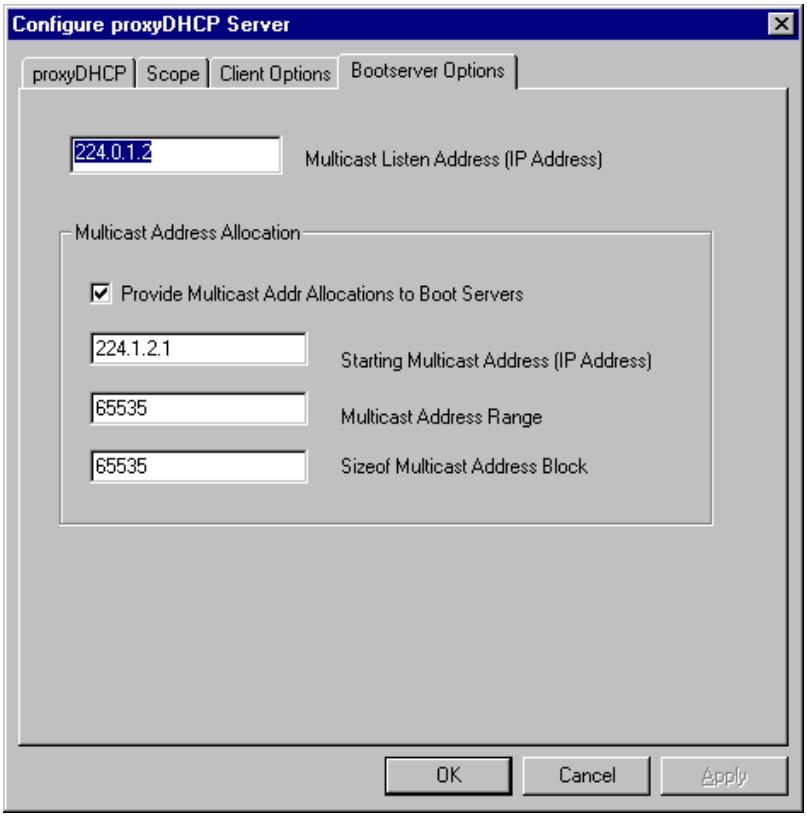
The first line of text for the booting client is also entered on this tab. This line of text appears on all boot menus regardless of client system architecture.
In addition, the Menu Prompt timeout is specified here. The timeout controls how long the boot menu is available before the PXE client defaults to the first Bootserver type in the list.

4.3.2.3.2 proxyDHCP to Bootserver Configuration Information

To effectively make use of Multicast Bootserver Discovery, it is necessary that all Bootservers in a given domain, regardless of type, be configured with the same multicast listening address. The booting client and the listening Bootserver share the requirement of knowing the multicast discovery address.
Insuring that all of the Bootservers in a given domain are configured with the same multicast discovery address can be done by hand (time consuming, messy, prone to error). Alternatively, proxyDHCP can be configured with this address and the Bootservers can query proxyDHCP to get it.
The Multicast Discovery listening address is defined on the third tab (Bootserver Options).



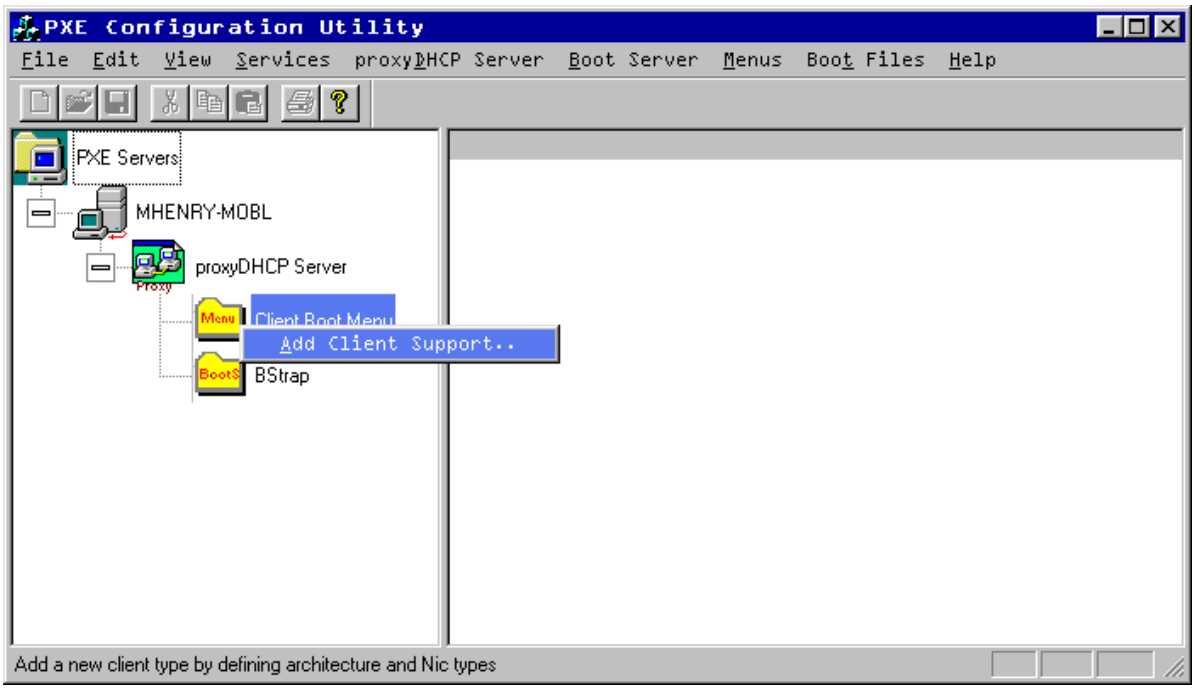
In addition, a block of multicast addresses can be made available for the Bootserver pool on this tab. If the Provide Multicast Address Allocation to Bootservers box is checked, then proxyDHCP will hand back to an inquiring Bootserver a block starting address, an Address Range parameter, and a size of address block parameter. proxyDHCP will configure inquiring Bootservers with a block of multicast addresses of Size of Multicast Address Block, with a start of block address randomly picked from within the Multicast Address Range.



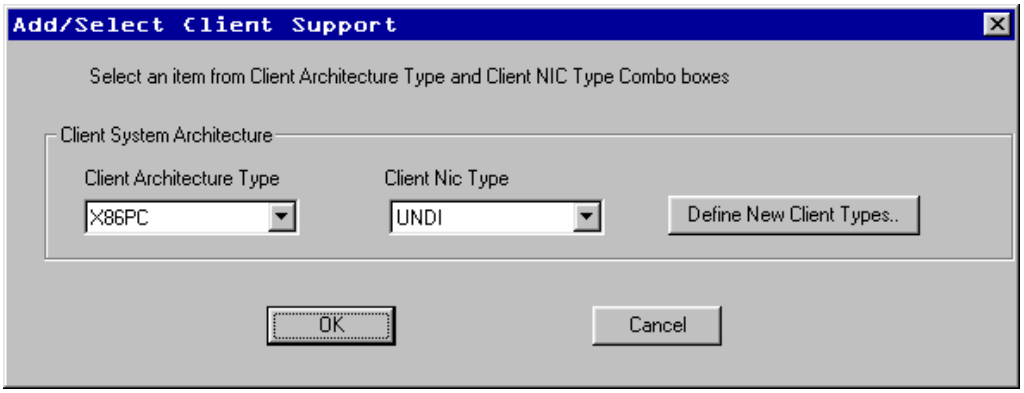
4.3.2.4 Client System Architecture Specific proxyDHCP Configuration Information

4.3.2.4.1 Define the System Architecture Type of Clients Supported

In this case, “support” means providing a Bootserver menu to the booting client. The proxyDHCP service can provide different boot menus for each system architecture client it supports. To create boot menus, it is first necessary to define the type of clients that are supported.

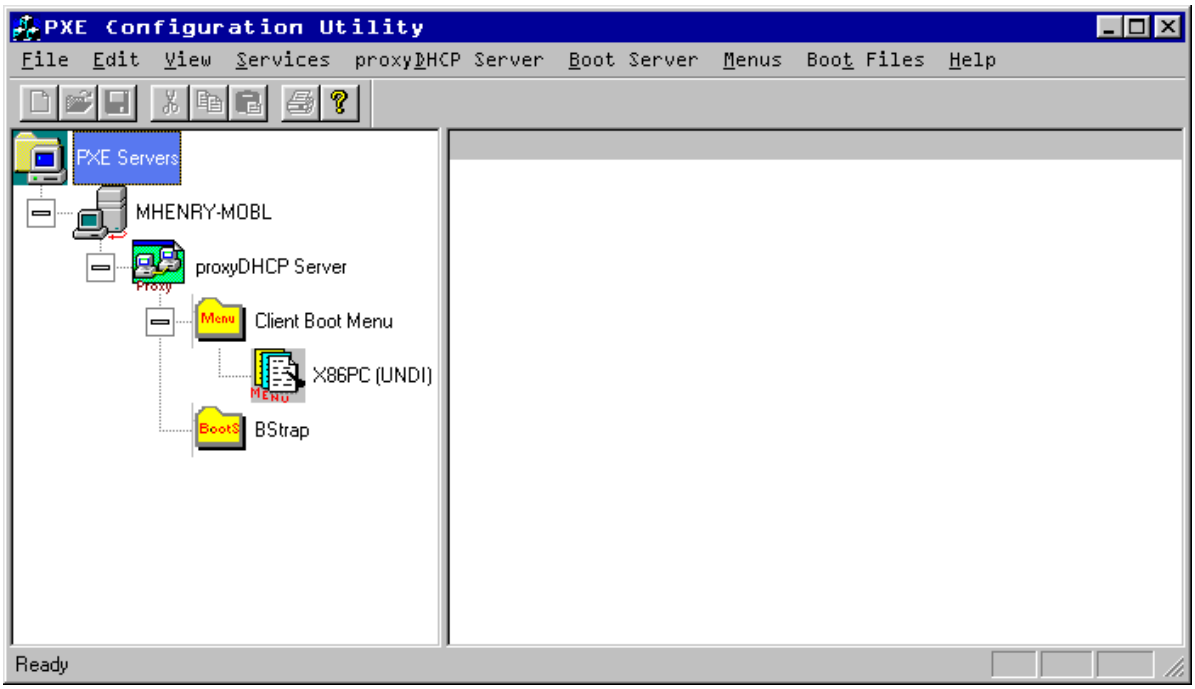


In addition to being able to select types from the Client System Architecture list defined in the PXE 2.0 specification, you may define your own type to provide support for a new system architecture not covered in the published list. (However, to get a permanent number assigned to a system architecture, the request must be made in writing via the Review Request form.)



In this case, a blank menu has been created for the X86PC architecture type in the Client Boot Menu folder.

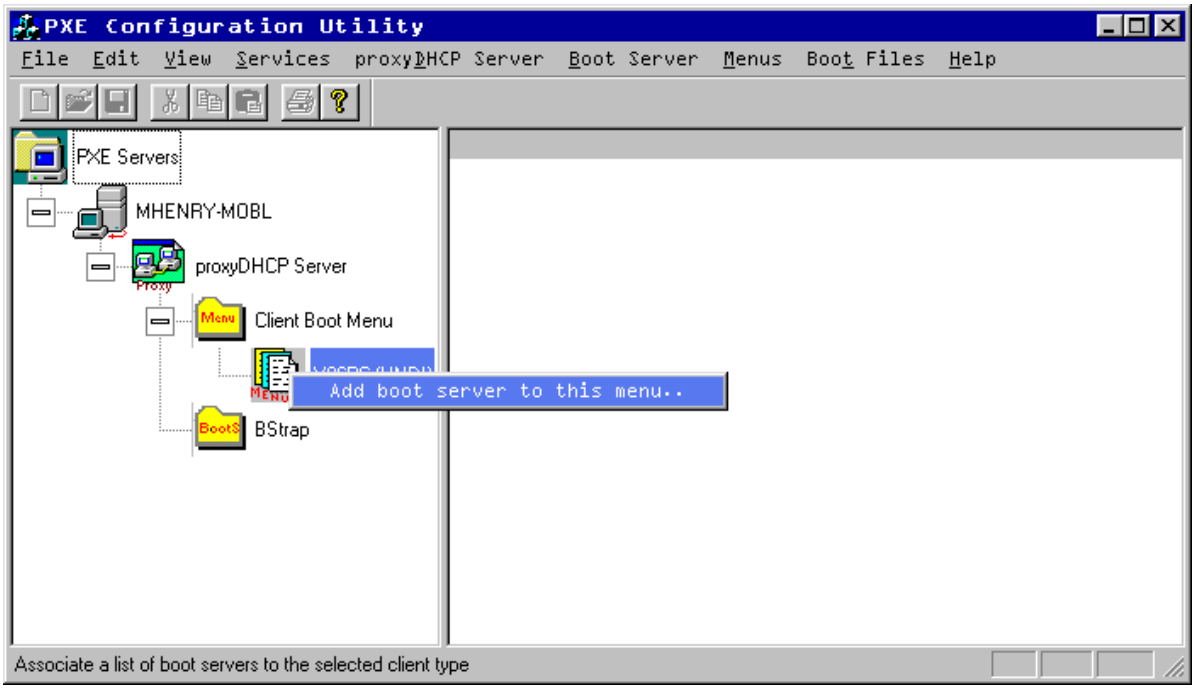
The next section defines how to create a list of Bootserver types to populate the menu.



4.3.2.4.2 Client Boot Menu(s)

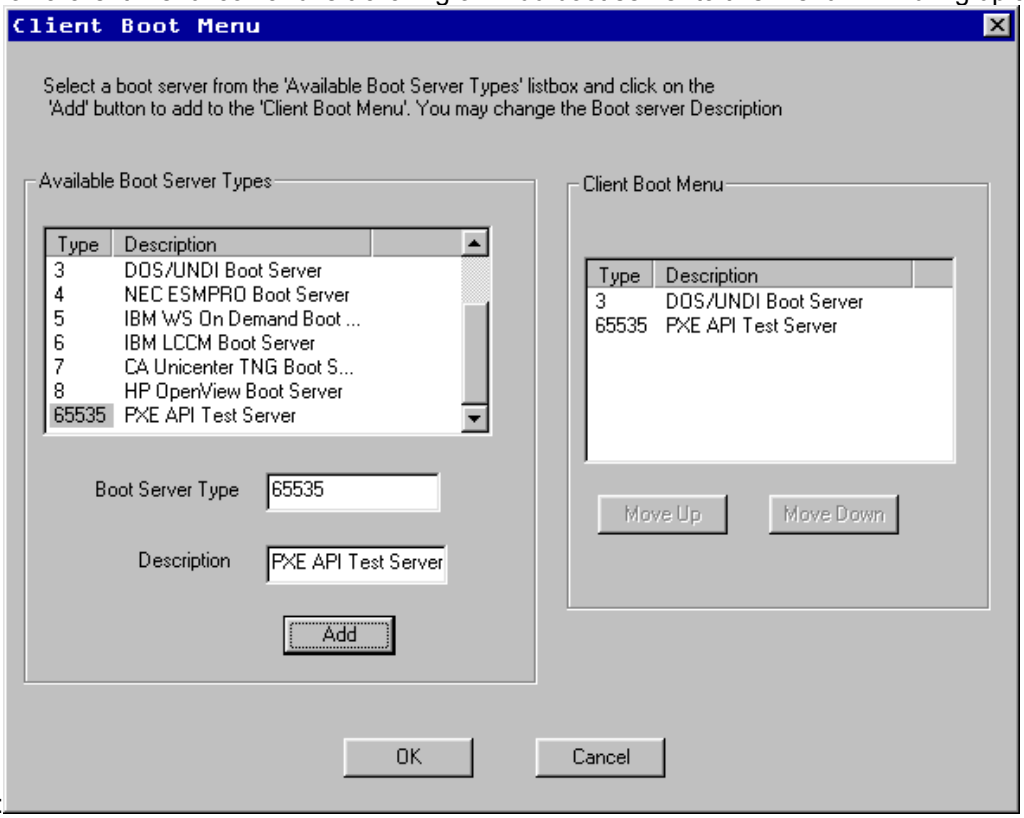
Client Boot Menus are created by defining a list of appropriate (and available) Bootservers that the client may want or need to boot from. This list can be any combination of Management and/or OS boot servers.

The first entry in the Bootserver list is the default device that the boot ROM will choose at the end of the time-out period if the user has not selected one of the other choices in the list.

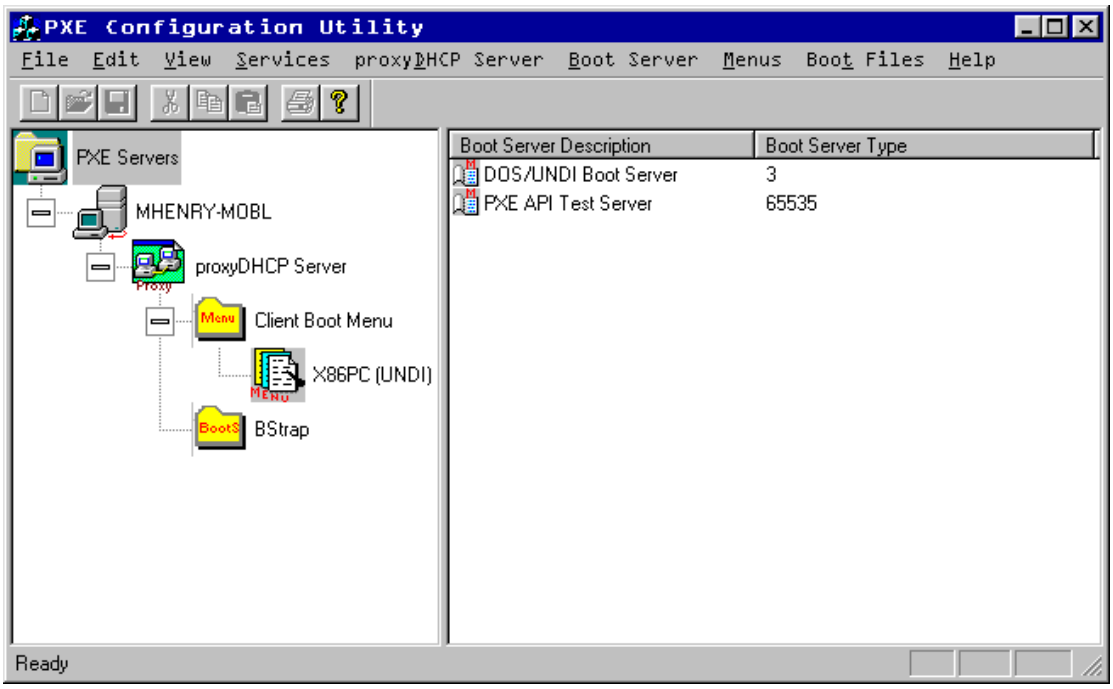


Again, it is possible to define new Bootserver types, with the same restrictions defined above for System Architecture types.

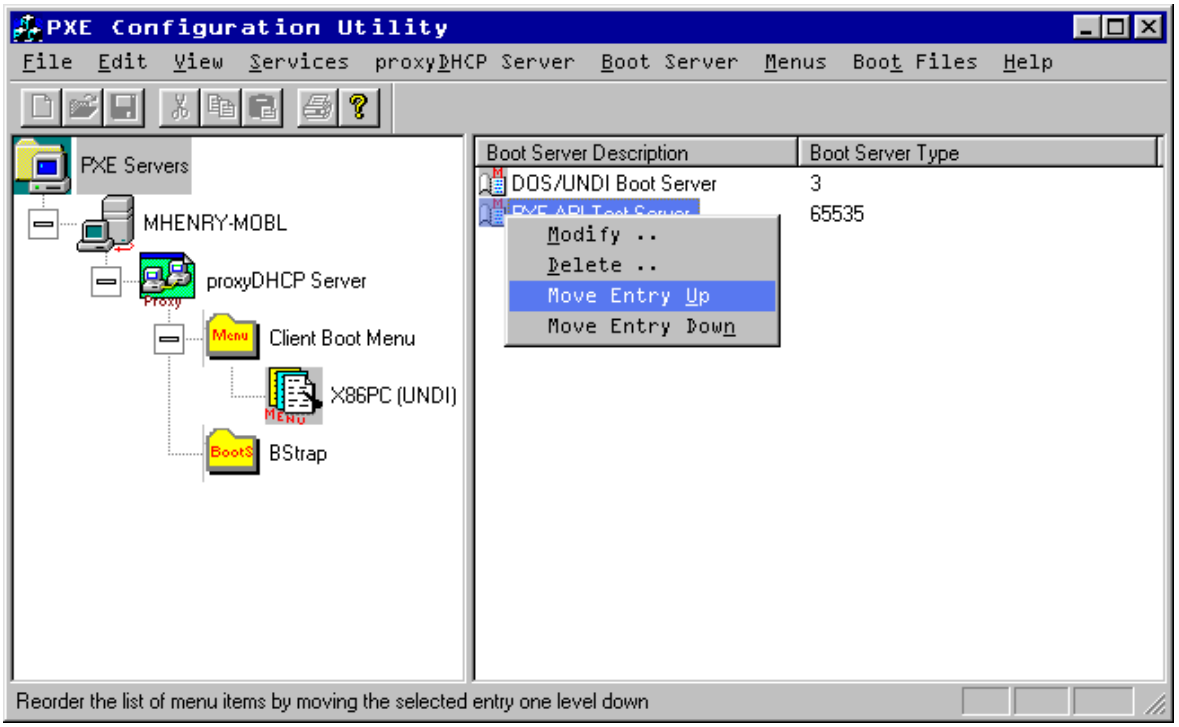
Right-clicking on a client menu icon and left-clicking on ‘Add boot server to this menu..’ will bring up the following box:

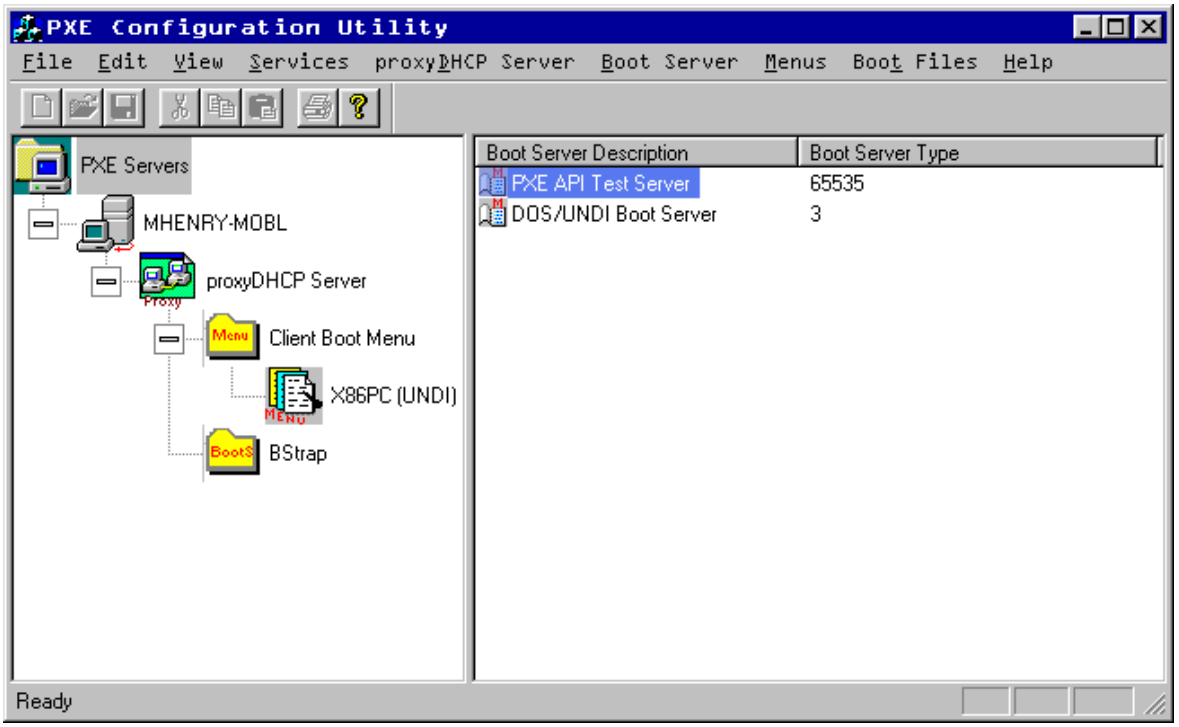


Here you can add items to the client menu, a list of available bootserver types. You can select from predefined types on the left-hand side list or just enter a new number and description and press add. The selected entries are shown on the right side. When you are done adding the required types, click on OK.



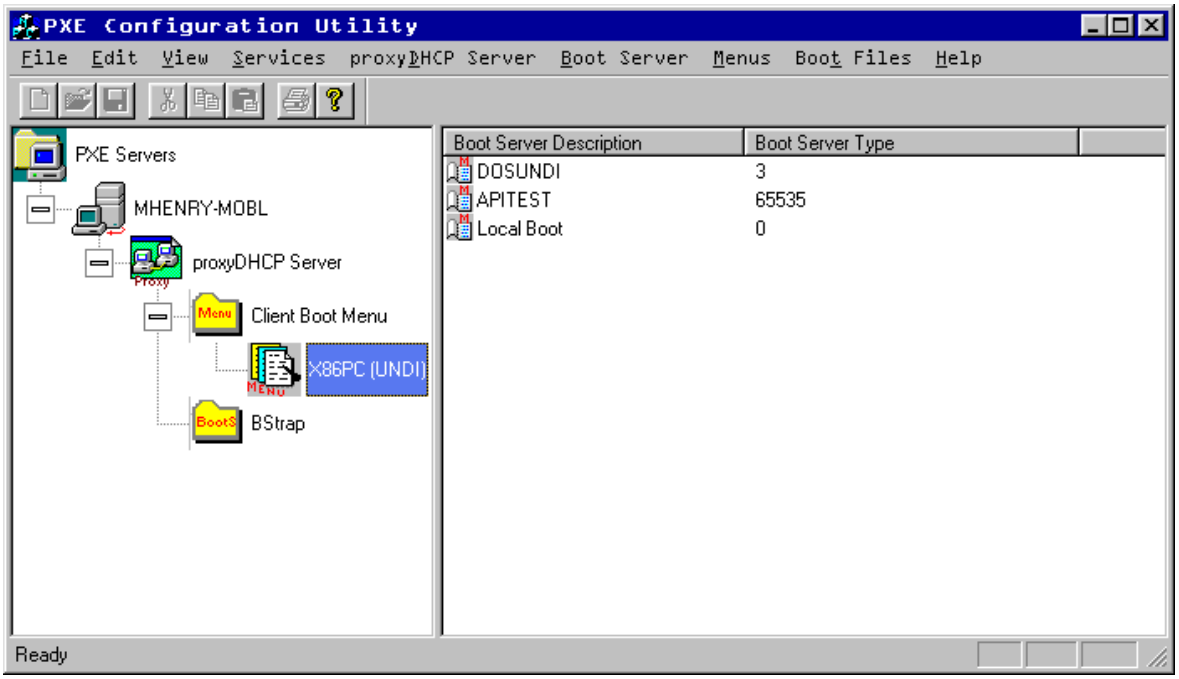
After selecting the Bootserver types, the order of the list can be rearranged if needed.





This PXE server is now configured to provide a Bootserver menu to X86PC-type clients. The Bootserver menu includes the two services provided in the PXE PDK: the first is a PXE boot ROM API test application boot server (this boot server is defined to do only a specific, restricted task); the second is a DOS OS boot server.

As mentioned earlier, Bootserver Type 0 is a reserved type for Local Boot. In plain language, using this type signals the PXE boot ROM to fail the boot. There are several potential uses for this type. For example, if the boot menu is defined dynamically (using a plug-in to the proxyDHCP service to determine which machine is booting), it would be possible to prevent unknown users from booting to any Bootserver, and at the same time record information about the booting client (UUID, architecture, sub-net, etc.) that would be useful for populating an inventory database.

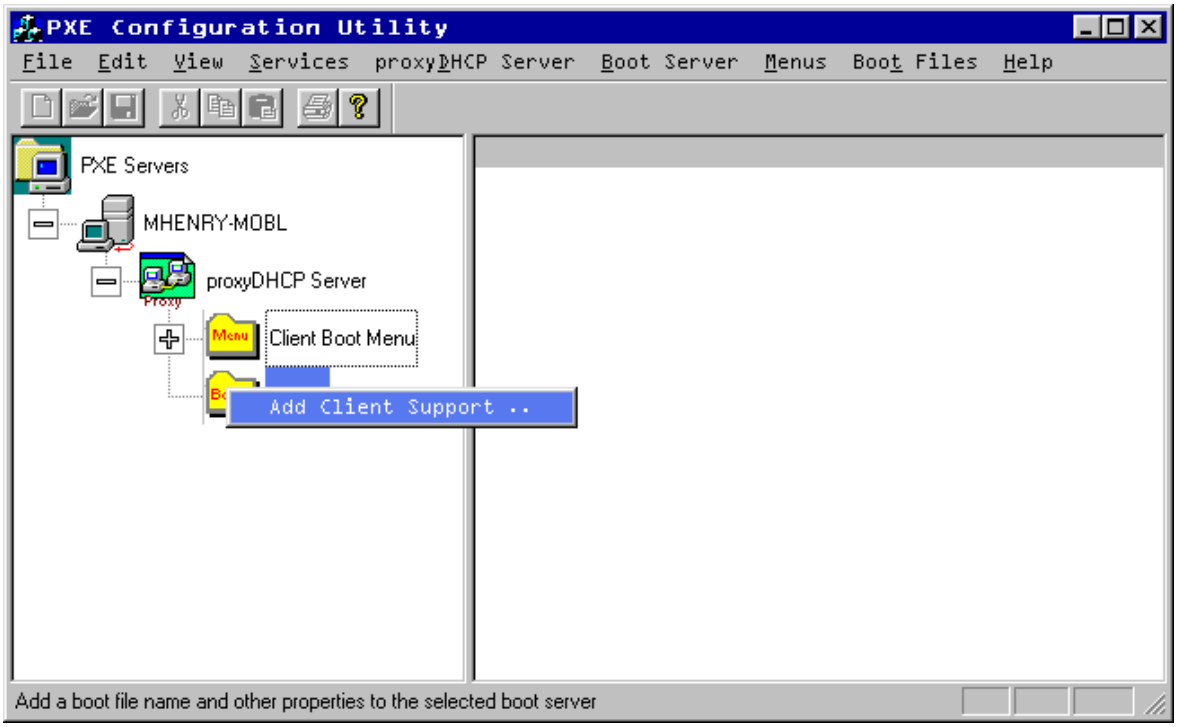


4.3.2.5 BSTRAP – Pre-PXE 2.0 Discovery Support

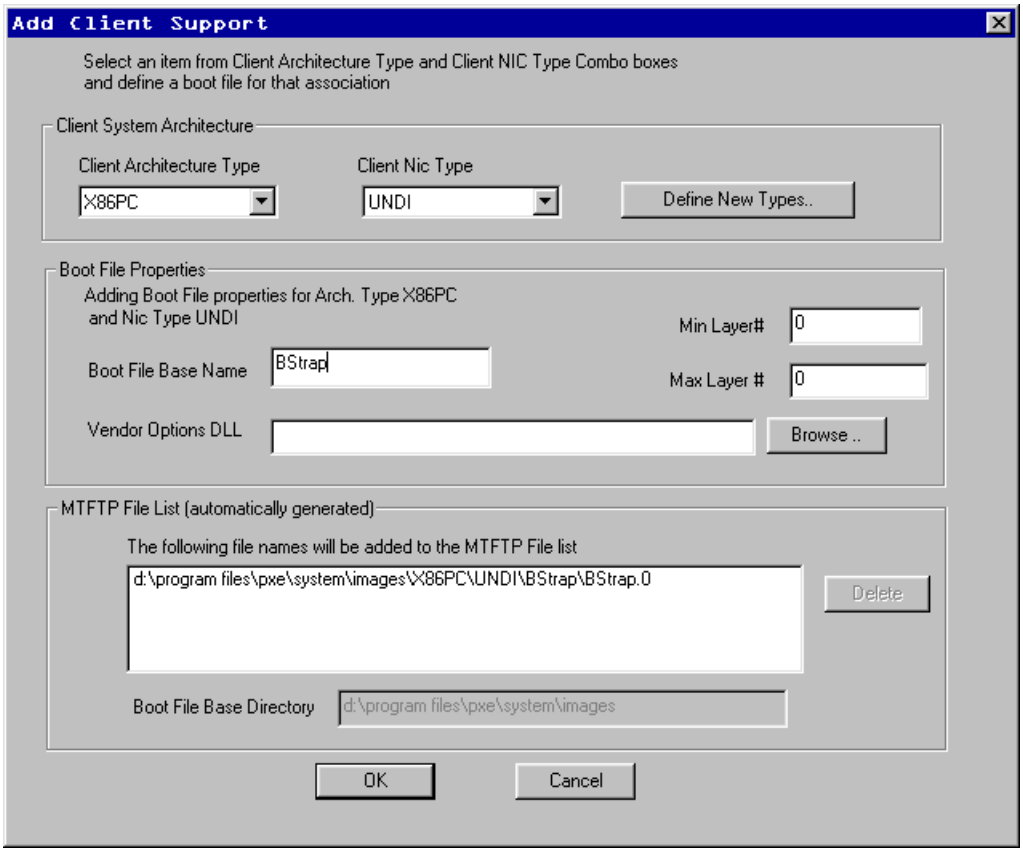
The BSTRAP (short for bootstrap) mechanism is intended to provide pre-PXE 2.0 clients with the ability to discover Bootservers. In essence, proxyDHCP acts as a mini-bootserver for v0.9x PXE clients by providing these clients a file containing the Bootserver discovery code. PXE 2.0 clients will ignore this offer and use their native discovery capability.

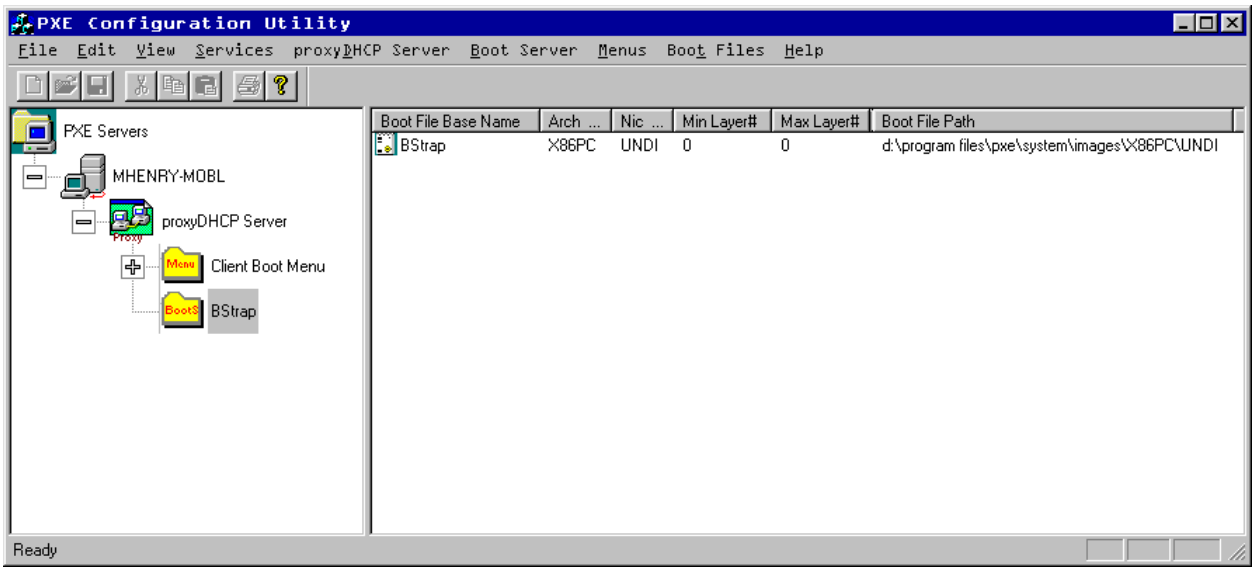
The BSTRAP file for X86PC architecture is contained in the PDK and is installed automatically at setup.

To configure this support, first define client types to be supported in the BSTRAP folder.



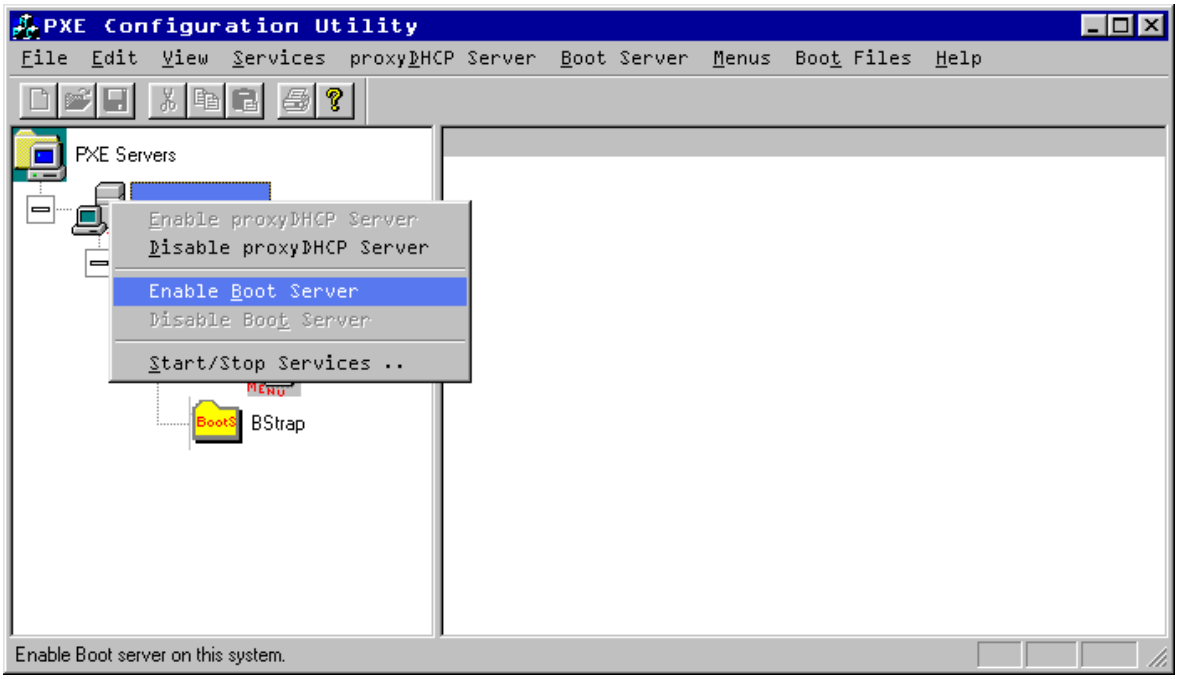
The configuration utility will define the directory for the BSTRAP file.



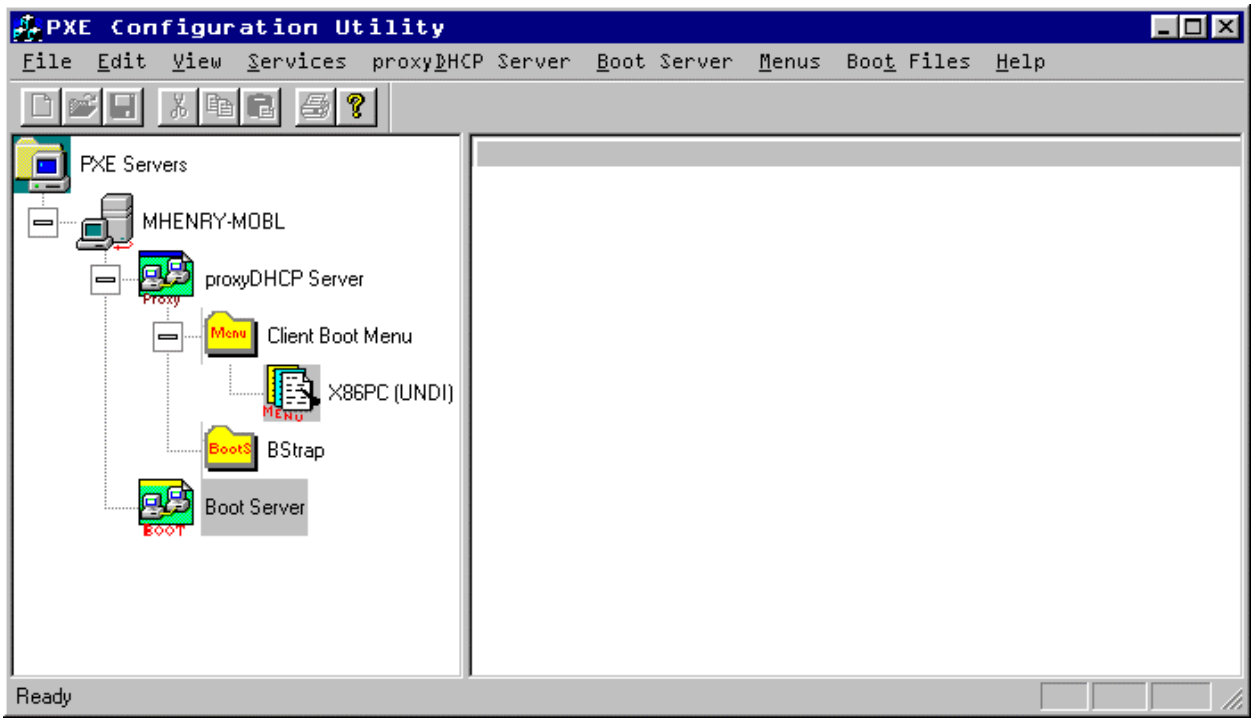


4.3.3 Configure Bootserver(s)

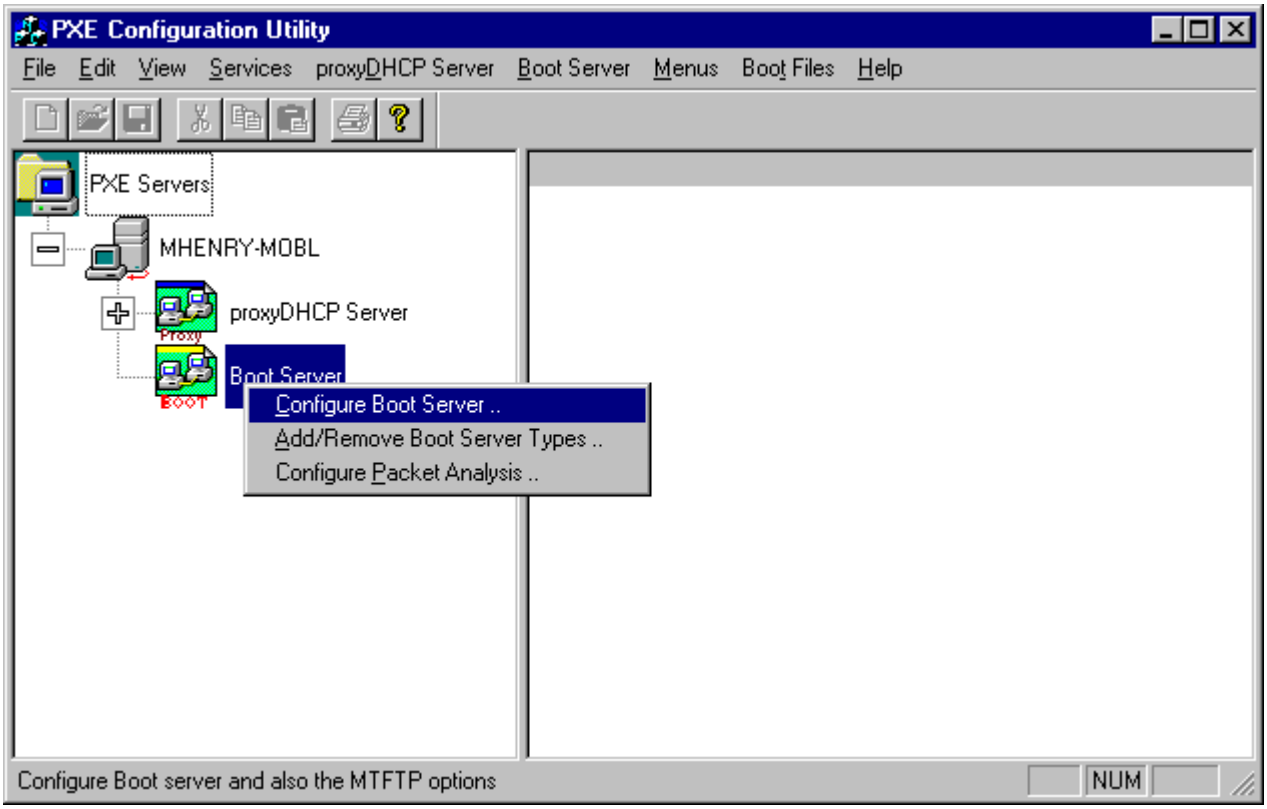
The Bootserver service is configured in a similar way as the proxyDHCP service. First the service must be enabled.



Selecting Enable Boot Server adds the Bootserver icon to the window.



Right-click the Bootserver icon.



Click Configure Boot Server to define global Bootserver configuration.

4.3.3.1 Global Bootserver Configuration Information

This dialog box is the mirror image of the proxyDHCP MTFTP address and UDP port configuration.

If the proxyDHCP and Bootserver are on the same machine, selecting UDP Port 67 in proxyDHCP automatically selects it in Bootserver.

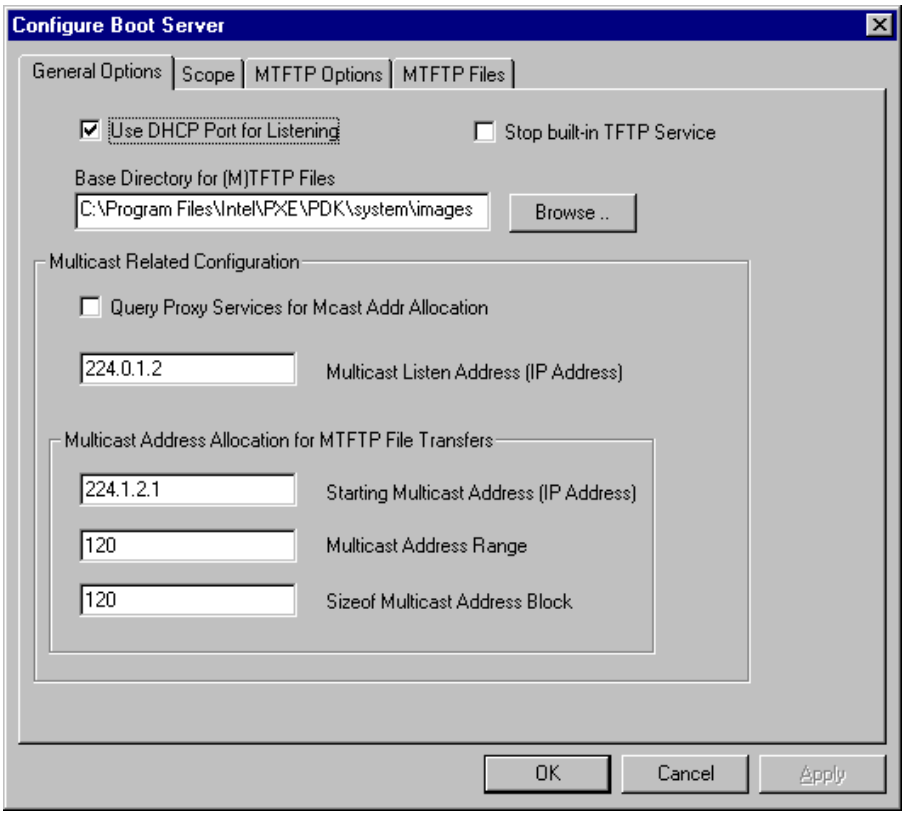
In the proxyDHCP case, the Multicast addresses were statically defined and either were or were not made available to inquiring Bootservers. In the Bootserver case, the addresses are statically defined, or the Bootserver is configured to get the addresses from the proxyDHCP server. This is also true for the multicast listening address (the address the booting client queries when using multicast discovery).

There are two ways for the Bootserver to get the multicast listening address. First, if both the proxyDHCP and Bootserver services are enabled on the same machine, then defining the multicast listening address for the proxyDHCP service automatically defines it for the Bootserver service.

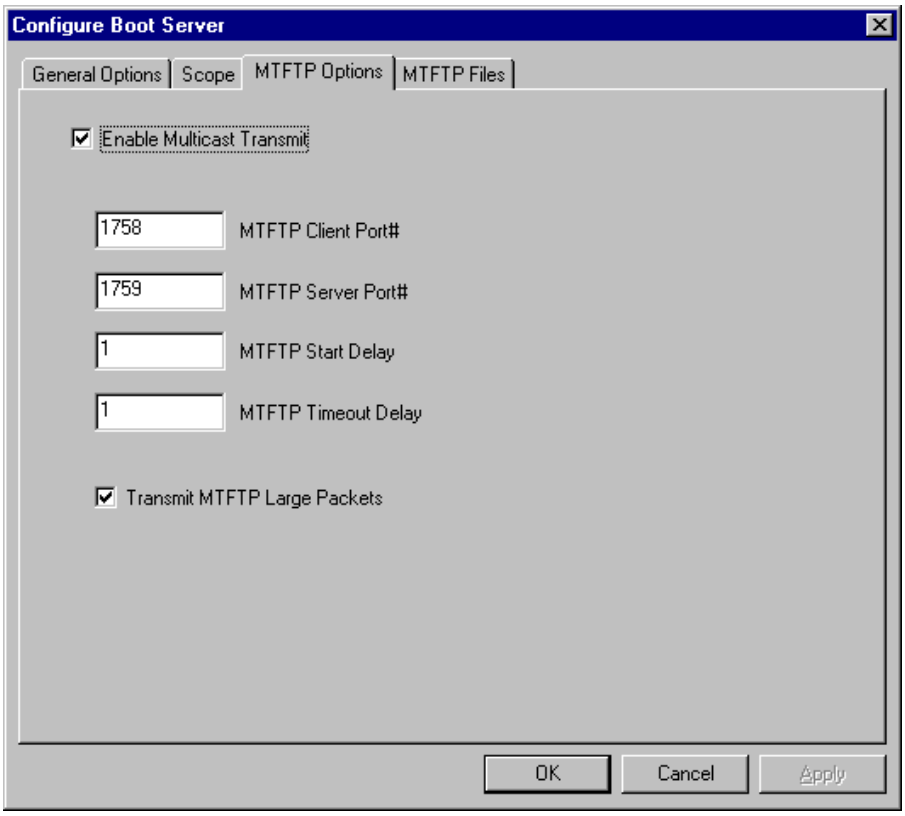
Second, if proxyDHCP and Bootserver are on different machines, and if the Query Proxy check box is selected, the Bootserver will query the proxyDHCP for the multicast listening address. Remember that in any case Multicast Address Range for the bootserver should be large enough to accommodate all of the MTFTP files, otherwise the MTFTP will not work (though TFTP will still work fine).

Please note that you have the option of using a third party TFTP server by stopping the built-in TFTP service using the shown check box. The Multicast TFTP (MTFTP) will still work. You can also disable Multicast transmit from the MTFTP Options tab and leave normal TFTP running.

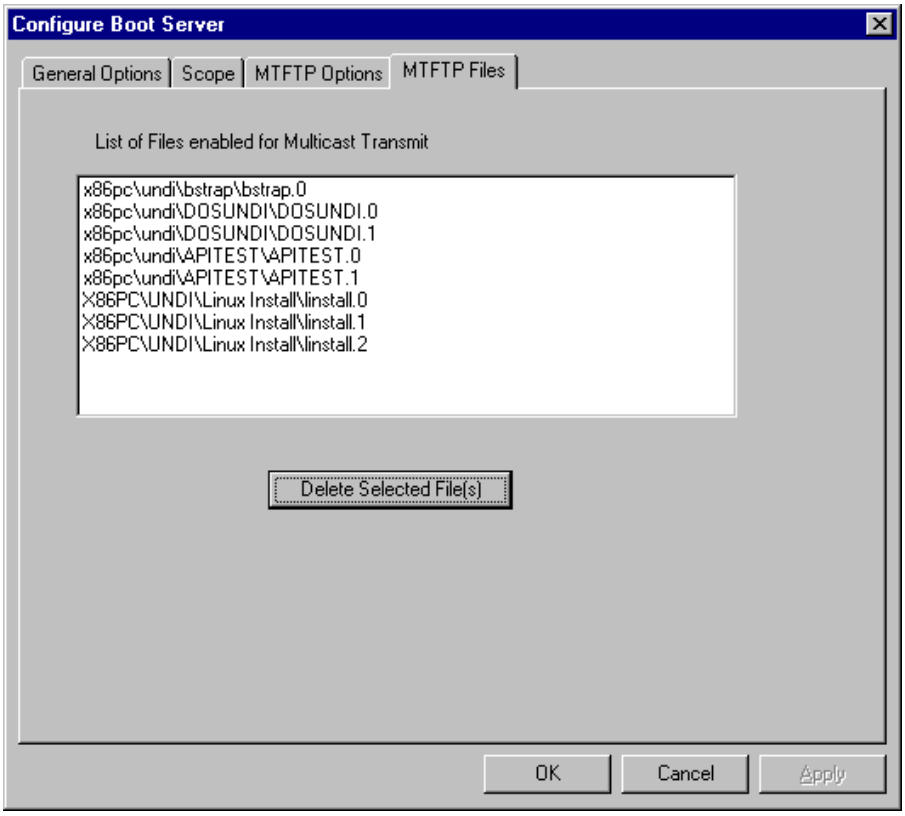
The Scope tab is exactly the same as for proxyDHCP.



The first checkbox in ‘MTFTP Options’ tab is to control the Multicast transmit only. This tab allows you to configure client and server port numbers. It also lets you enable transmission of large packets through MTFTP.

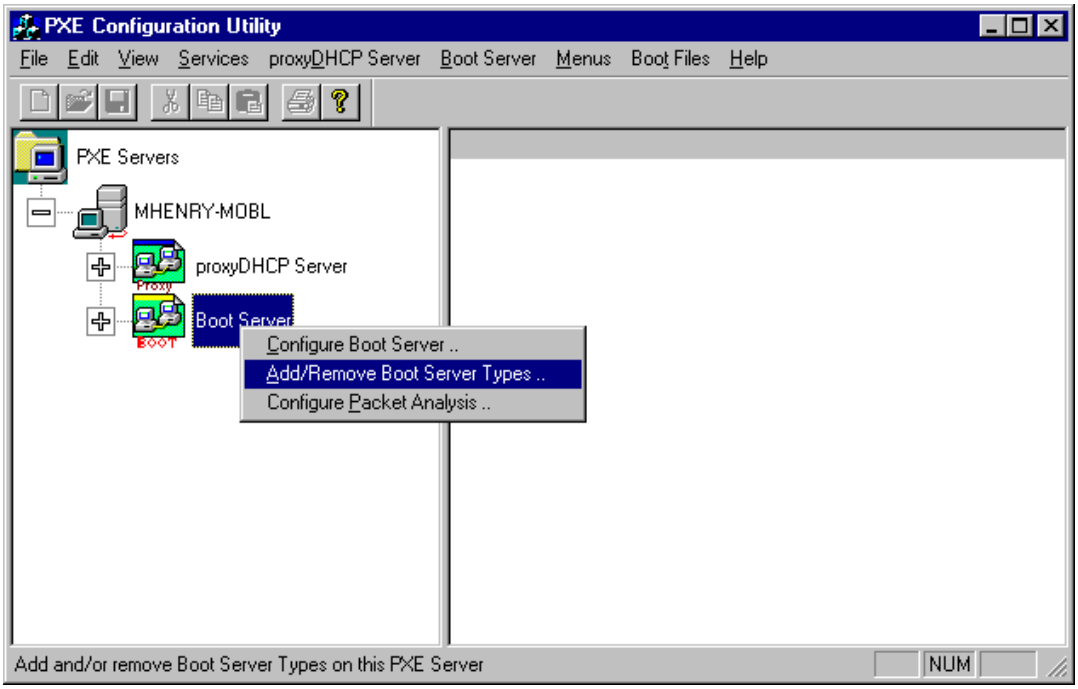


The last tab shows the files that are enabled for multicast transmit. Each file gets assigned a multicast IP address when the PXE MTFT service starts up. The IP address depends on the multicast start address and the range. You should manage this list periodically to remove excess files.

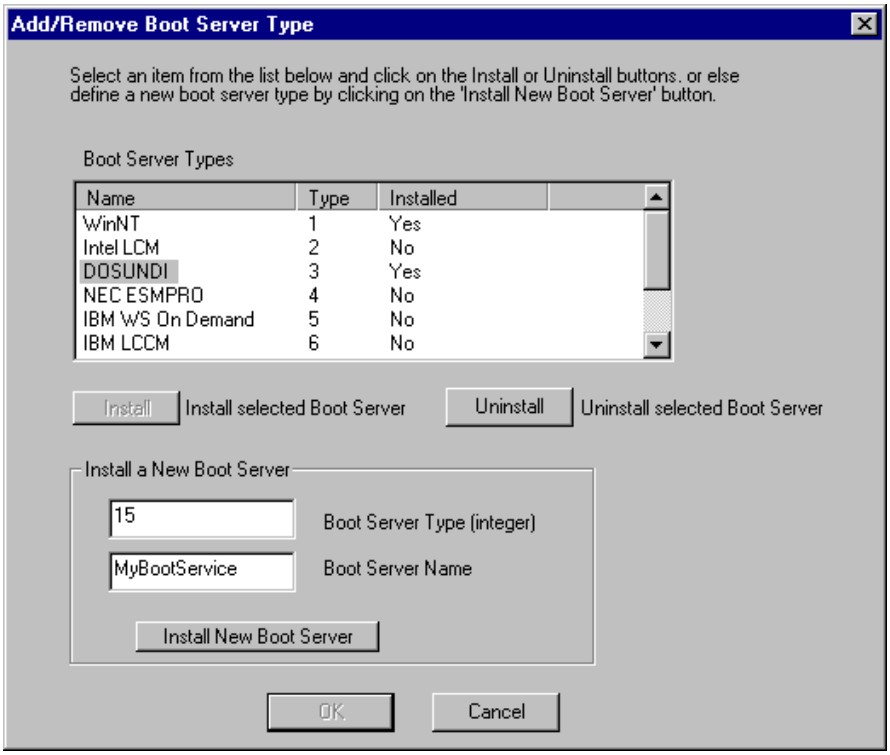


4.3.3.2 Installing Boot Servers

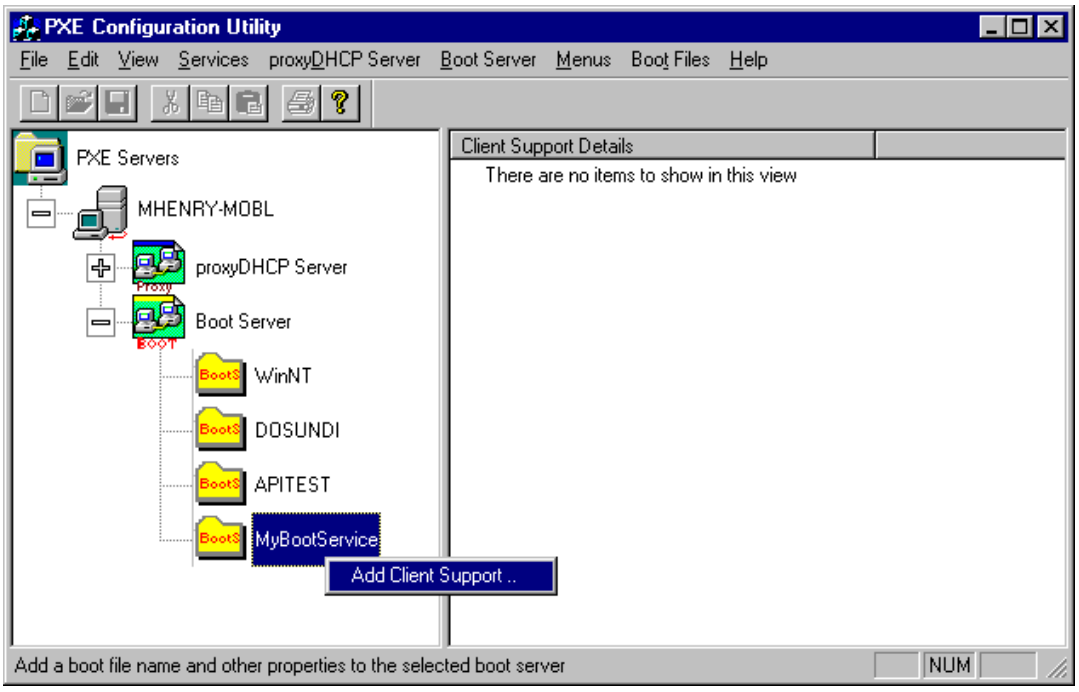
The PXE setup gives some pre-installed boot servers from the list of standard bootserver types. The installed standard bootserver types are WinNT, DOSUNDI, APITEST and RIStrap.
To modify bootservers, click on ‘Add/Remove Boot Server Types…’ as shown below.



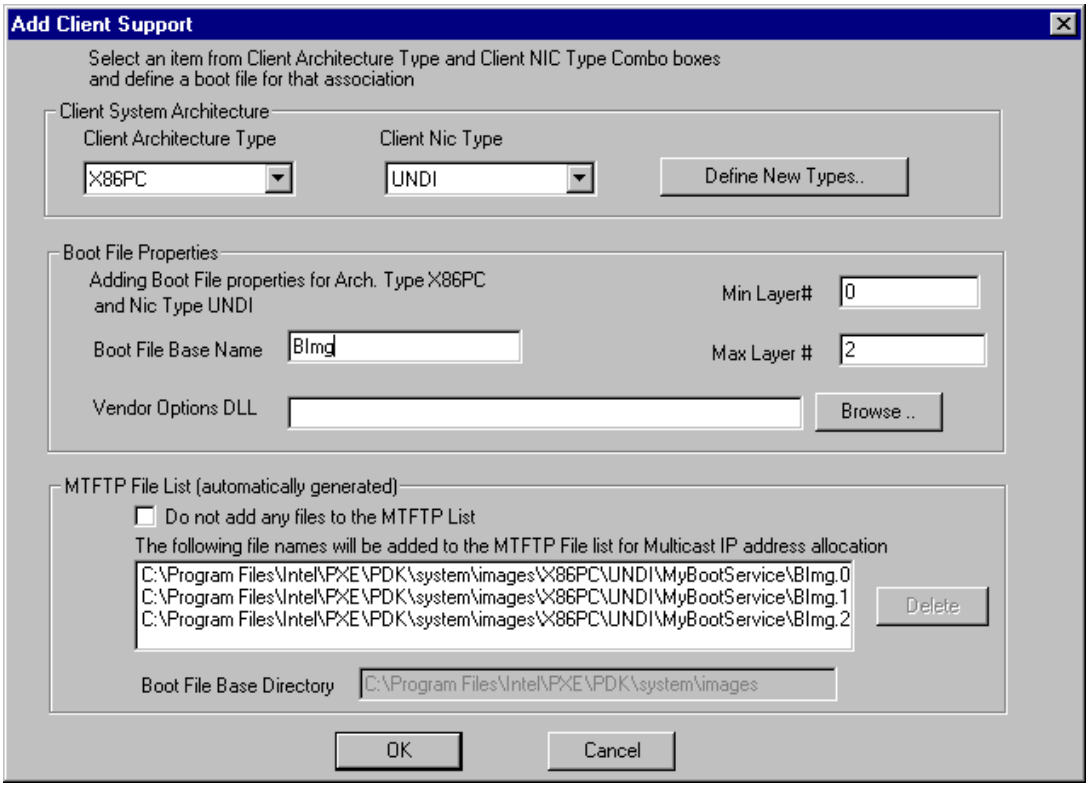
Here you can install a predefined or standard bootserver or a new bootserver using an unused type number as shown.



The bootserver type selected for installation will appear in the ‘Boot Server’ subtree as shown below. To add further information to the particular bootserver, right click on it and select ‘Add Client Support’.



Here you will provide the boot image file information. The layer # refer to number of image files that are associated with this bootserver. These files should be present at the path shown in the bottom list.



After you click ‘OK’, you will be able to see an entry on the right side for the particular boot server on the main window. If you wish, you can now add this bootserver type to your proxyDHCP menu.

4.4 Registry Entries

The following table lists the registry entries that the PXE Configuration Utility manages.

Registry Keys for PXE Services	
Key Name: SOFTWARE\Intel\PXE	This is the root registry key for all PXE Services.
Value 1 Name: PxeMtftp_Debug_FileName Type: REG_SZ Data: C:\PxeMtftpDebugLog.Txt	Where MTFTP service debug information will be written, if debugging to a file is enabled.
Value 2 Name: PxeMtftp_DebugOutToFile_On Type: REG_DWORD Data: 0	Set to 1 to enable MTFTP service debugging to a file.
Value 3 Name: PxeMtftp_DebugOutToWindow_On Type: REG_DWORD Data: 0	Set to 1 to enable MTFTP service debugging to a window.
Value 4 Name: PxeParsers_Debug_FileName Type: REG_SZ Data: C:\PxeParsersDebugLog.Txt	Where PXE parser service debug information will be written, if debugging to a file is enabled.
Value 5 Name: PxeParsers_DebugOutToFile_On Type: REG_DWORD Data: 0	Set to 1 to enable PXE parser debugging to a file.
Value 6 Name: PxeParsers_DebugOutToWindow_On Type: REG_DWORD Data: 0	Set to 1 to enable PXE parser debugging to a window.
Value 7 Name: PxeServices_Debug_FileName Type: REG_SZ Data: C:\PxeServicesDebugLog.Txt	Where PXE service debug information will be written, if debugging to a file is enabled.
Value 8 Name: PxeServices_DebugOutToFile_On Type: REG_DWORD Data: 0	Set to 1 to enable PXE service debugging to a file.
Value 9 Name: PxeServices_DebugOutToWindow_On Type: REG_DWORD Data: 0	Set to 1 to enable PXE service debugging to a window.
Value 10 Name: PxeTester_Debug_FileName Type: REG_SZ Data: C:\PxeTesterDebugLog.Txt	Where PXE client test results will be written, if writing to a file is enabled.
Value 11 Name: PxeTester_DebugOutToFile_On Type: REG_DWORD Data: 0	Set to 1 to enable writing client test results to a file.
Value 12 Name: PxeTester_DebugOutToWindow_On Type: REG_DWORD Data: 0	Set to 1 to enable writing client test results to a window.
Key Name: SOFTWARE\Intel\PXE\PDK\<build_number>	Installed PDK version number.
MTFTPD	
Key Name: SOFTWARE\Intel\PXE\MTFTPD	Top Level for MTFTPD service. Contains one sub-key, FILES.

Registry Keys for PXE Services	
Value 0 Name: BASE_DIR Type: REG_SZ Data: <install directory>SYSTEM\IMAGES	The base directory where bootfiles are located. This path is pre-pended to the partial path that the client requests.
Value 1 Name: MCAST_ADDRESS_RANGE Type: REG_SZ Data: 7	Range of the IP addresses assigned to the boot server.
Value 2 Name: MCAST_BASE_ADDRESS Type: REG_SZ Data: 224.1.5.1	The starting multicast IP address of the range of addresses assigned to the files.
Value 3 Name: MCAST_BLOCK_SIZE Type: REG_SZ Data: 10	Size of the block of IP addresses assigned to the boot server.
Value 4 Name: MCAST_ENABLE Type: REG_DWORD Data: 0x1	Determines whether multicast is enabled. 1 => Multicast Enabled 2 => Multicast Disabled
Value 5 Name: MTFTP_HIGH_PERFORMANCE Type: REG_DWORD Data: 0	Enables the use of large packets in file transfers. 1 => Enabled 0 => Disabled
Value 6 Name: Service_Name Type: REG_SZ Data: Intel PXE MTFTP Service	Name of the MTFTPD service.
Value 7 Name: Service_Path Type: REG_SZ Data: <install directory>system\PXEMTFTP.exe	Location of the MTFTP service.
Key Name: SOFTWARE\Intel\PXE\MTFTPD\FILES	Contains a set of registry values for each file to be sent by MTFTPD. The data for each value is the multicast IP address for the file. These addresses are computed automatically, starting with MCAST_BASE_ADDRESS, when the PXE services are started.
Value 0 Name: X86PC\UNDI\APITEST\APITEST.0 Type: REG_SZ Data: 224.1.5.4	First bootfile for the API Test.
Value 1 Name: X86PC\UNDI\APITEST\APITEST.1 Type: REG_SZ Data: 224.1.5.5	Second bootfile for the API Test.
Value 2 Name: X86PC\UNDI\BSTRAP\BSTRAP.0 Type: REG_SZ Data: 224.1.5.1	Bootstrap program that is sent to all WfM 1.1a (PXE 0.99) boot ROMs. This bootstrap implements PXE 2.0 menuing and discovery.
Value 3 Name: X86PC\UNDI\DOSUNDI\DOSUNDI.0 Type: REG_SZ Data: 224.1.5.2	First bootfile for the DOS boot.
Value 4 Name: X86PC\UNDI\DOSUNDI\DOSUNDI.1 Type: REG_SZ Data: 224.1.5.3	Second bootfile for the DOS boot.
proxyDHCP	
Key Name: SOFTWARE\Intel\PXE\proxyDHCP	Top level for the proxyDHCP service. Contains sub-keys for each PC architecture supported.
Value 0 Name: Discovery_BCast_Disabled Type: REG_DWORD Data: 0	Set to 1 to disable PXE client broadcast discovery.
Value 1 Name: Discovery_List Type: REG_MULTI_SZ Data:	This the list of bootserver types and IP addresses that the client will do unicast discovery to. Also, if Discovery_Server_List_Only is set to 1, the client will only accept responses from servers in this list.

Registry Keys for PXE Services	
Value 2 Name: Discovery_Local_Bstrap Type: REG_DWORD Data: 0	If this value is set to 1, then the client has to discover boot server type 0 and download. Otherwise, it can use its built-in one.
Value 3 Name: Discovery_MCast_Addr Type: REG_SZ Data: 224.0.1.2	This is the PXE bootserver multicast listening address. All PXE bootservers on a network should have the same multicast discovery address.
Value 4 Name: Discovery_MCast_Disabled Type: REG_DWORD Data: 0x1	Disable multicasting discovery packets. 1 => multicasting disabled 0 => multicasting enabled
Value 5 Name: Discovery_Server_List_Only Type: REG_DWORD Data: 0	Enable server list. If enabled client is to only accept bootfiles from the list of servers contained in the Discovery_List registry value. 0 => Disable Discovery_List 1 => Enable Discovery_List
Value 6 Name: Discovery_Srvr_IP Type: REG_SZ Data: 0.0.0.0	Unicast IP address of the discovery server that provides BStrap.0.
Value 7 Name: Domain Type: REG_SZ Data: PXE	Domain or Workgroup name for this server.
Value 8 Name: IsDomain Type: REG_DWORD Data: 0	Indicates whether the Domain name reflects a Workgroup or a Domain.
Value 9 Name: MasterProxy Type: REG_DWORD Data: 1	Bit indicating whether the boot server should query a proxyDHCP for Multicast address allocation or use its own registry.
Value 10 Name: Parsers Type: REG_MULTI_SZ Data: PXEClientTester PxeParser ConfigServer	List of plug-ins that the PxeService uses to process incoming packets.
Value 11 Name: PROC_ARCH Type: REG_MULTI_SZ Data: 0,X86PC	Mapping of Client System Architecture types to sub-key names.
Value 12 Name: Prompt Type: REG_SZ Data: 10,Press F8 to view menu.	Text and display time for the prompt displayed after the client menu.
Value 13 Name: ServerName Type: REG_SZ Data: PXESERVER	Computer name of this server.
Value 14 Name: Service_Types Type: REG_MULTI_SZ Data: 0,Bstrap 1,WinNT 3,DOSUNDI 65535,APITEST	Mapping of Bootserver types to sub-key and directory names.
Value 15 Name: StartBootService Type: REG_DWORD Data: 0x1	Enables the boot server.
Value 16 Name: StartProxy Type: REG_DWORD Data: 0x1	Enables the proxyDHCP server.
Value 17 Name: TestOn Type: REG_DWORD Data: 0	Output packet analysis for client testing.

Registry Keys for PXE Services	
Value 18 Name: UseDHCPPort Type: REG_DWORD Data: 0x1	Controls whether the PXE Bootserver listens on port 67. This should be turned off if PXE Bootserver shares the server with a DHCP service.
Value 19 Name: StopTftp Type: REG_DWORD Data: 0x0	Disable the TFTP part of PXE [M]TFTP service. 0 = TFTP enabled. 1 = TFTP disabled.
Key Name: SOFTWARE\Intel\PXE\proxyDHCP\Parsers	Top level for Parser entries. Contains a sub-key for each parser installed.
Key Name: SOFTWARE\Intel\PXE\proxyDHCP\Parsers\ ConfigServer	Contains a set of registry values for the PxeServerParser that provides configuration services for the boot server regarding multicast address allocation.
Value 0 Name: Discovery_MCast_Addr Type: REG_SZ Data: 224.0.1.2	The multicast address for sending multicast discovery packets.
Value 1 Name: DllPath Type: REG_SZ Data: <install directory>system\PXEServerParser.dll	Path where the DLL can be found.
Value 2 Name: MainFunction Type: REG_SZ Data: PXEServerParser	Entry point for the parser.
Value 3 Name: Mcast_Allocation_Address Type: REG_SZ Data: 224.1.2.1	First multicast IP address that can be allocated and handed out to slave proxydhcp servers.
Value 4 Name: Mcast_Allocation_Block Type: REG_SZ Data: 65535	Size of multicast IP address block that can be handed out to a slave proxydhcp server.
Value 5 Name: Mcast_Allocation_Last Type: REG_SZ Data: 0.0.0.0	Last multicast IP address that was handed out.
Value 6 Name: Mcast_Allocation_Range Type: REG_SZ Data: 65535	Total number of multicast IP addresses that can be handed out.
Key Name: SOFTWARE\Intel\PXE\proxyDHCP\Parsers\ PXEClientTester	Contains a set of registry values for the PxeClientTest parser that supports APITest.
Value 0 Name: DllPath Type: REG_SZ Data: <install directory>system\PXEClientTester.dll	Path where the DLL can be found.
Value 1 Name: MainFunction Type: REG_SZ Data: PXEClientTester	Entry point for the parser.
Key Name: SOFTWARE\Intel\PXE\proxyDHCP\Parsers\ PxeParser	Contains a set of registry values for the PxeParser that provides the main packet parsing functionality for the proxyDHCP and boot servers.
Value 0 Name: DllPath Type: REG_SZ Data: <install directory>system\PxeParser.dll	Path where the DLL can be found.
Value 1 Name: MainFunction Type: REG_SZ Data: PxeParser	Entry point for the parser.
Key Name: SOFTWARE\Intel\PXE\proxyDHCP\X86PC\UNDI\ BStrap	Contains data for telling the client how to download the bootfiles for the API Test Bootserver.

Registry Keys for PXE Services	

Value 0
  Name: Imagefile_Name
  Type: REG_MULTI_SZ
  Data: 0
  0
  Bstrap
	Relative path to the BStrap.0.

Value 1
  Name: Vendor_DLL
  Type: REG_SZ
  Data: <install directory>system\BstrapOpts.Dll
	<RESERVED> Will point to a DLL that can provide any vendor options that are unique to this Bootserver.

Value 2
  Name: Vendor_Options
  Type: REG_BINARY
  Data:
	<RESERVED> Will provide the ability to specify vendor options unique to this Bootserver directly in the registry.

Key Name: SOFTWARE\Intel\PXE\proxyDHCP\X86PC\UNDI	Key to contain data unique to X86Pcs with NICs that have UNDI interfaces.

Value 0
  Name: MENU
  Type: REG_MULTI_SZ
  Data: 0,Local Boot
  3,DOSUNDI
  65535,APITEST
	Menu list displayed by BStrap.0 on the client. Each entry is the Bootserver number followed by the menu item text, separated by a comma. Each menu item is a separate string.

Key Name:SOFTWARE\Intel\PXE\proxyDHCP\X86PC\UNDI\APITest	Contains data for telling the client how to download the bootfiles for the API Test Bootserver.

Value 0
  Name: Imagefile_Name
  Type: REG_MULTI_SZ
  Data: 0
  1
  APITest
	Root name of the bootfile along with the low and high valid index numbers.

Value 1
  Name: Vendor_DLL
  Type: REG_SZ
  Data: <install directory>system\ApiTestOpts.Dll
	<RESERVED> Will point to a DLL that can provide any vendor options that are unique to this Bootserver.

Value 2
  Name: Vendor_Options
  Type: REG_BINARY
  Data:
	<RESERVED> Will provide the ability to specify vendor options unique to this Bootserver directly in the registry.

Key Name:SOFTWARE\Intel\PXE\proxyDHCP\X86PC\UNDI\DOSUNDI	Contains data for telling the client how to download the bootfiles for the DOS UNDI Bootserver.

Value 0
  Name: Imagefile_Name
  Type: REG_MULTI_SZ
  Data: 0
  1
  DOSUNDI
	Root name of the bootfile.

Value 1
  Name: Vendor_DLL
  Type: REG_SZ
  Data: <install directory>system\DosUndiOpts.Dll
	<RESERVED> Will point to a DLL that can provide any vendor options that are unique to this Bootserver.

Value 2
  Name: Vendor_Options
  Type: REG_BINARY
  Data:
	<RESERVED> Will provide the ability to specify vendor options unique to this Bootserver directly in the registry.

Key Name: SOFTWARE\Intel\PXE\proxyDHCP\Scope	Top level key for all scope definitions.

Value 0
  Name: ServeAll
  Type: REG_SZ
  Data: 0
	Scoping bypass key.

Key Name: SOFTWARE\Intel\PXE\proxyDHCP\Scope\<name>	Contains a set of registry values for a user defined <name> scope. <name> is replaced with the name of the user defined Scope.

Value 0
  Name: Active
  Type: REG_SZ
  Data: 1
	Determines whether scope is active or inactive.

Registry Keys for PXE Services	

Value 1
  Name: ClassID PXEClient
  Type: REG_SZ
  Data:
	A user defined Type – Value pair content for <name> Scope.

5. Testing PXE

5.1 Tests Provided by PDK

Three different types of tests are provided in this PDK:

1. Packet Analysis
2. UNDI Stress Test
3. PXE API Tests

5.1.1 Packet Analysis

DHCP packet analysis consists of determining if the packet correctly contains the following options:

Description	Option Number
Client machine identifier (UUID)	97
Client system architecture option	93
Client network interface identifier	94
Correct DHCP message type	53
Class Identifier set to “PXEClient:Arch:xxxxx:UNDI:yyyzzz”	60

If any of these options are missing or the DHCP message-type (option 53) and/or class-identifier (option 60) contain incorrect values the client is considered to fail the compliance test.

5.1.1.1 Enabling Packet Analysis Test on the proxyDHCP Server

By default, this feature is turned off in the proxyDHCP server. To enable this feature, use regedit.exe to set the DWORD value TestOn under the registry key HKEY_LOCAL_MACHINE\SOFTWARE\Intel\PXE\ProxyDhcp to 1 and restart the services. The same thing can also be achieved using the PXE Configuration Utility program. After launching the configuration utility, right-click the proxyDHCP server or Bootserver. Then, select Configure Packet Analysis APITest ... to set the test to ON.

5.1.2 UNDI/NDIS Stress Test

To test the UNDI API with the universal NDIS driver, copy files with different sizes (maximum file size is 5MB) from the server's mapped drive to the RAMDisk, and then compare the two sets of files.

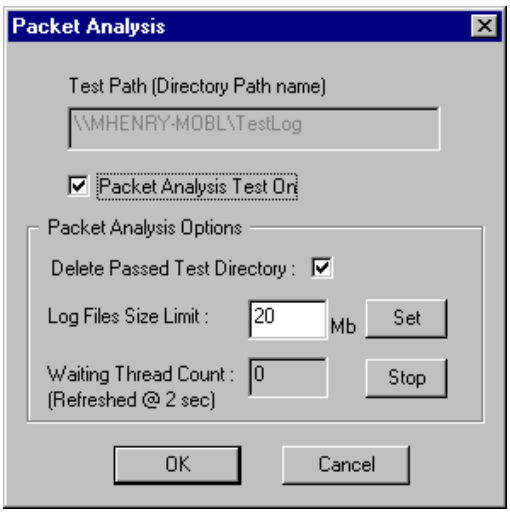
5.1.3 PXE API Test

The PXE specification contains a list of API calls that must be implemented by a PXE boot ROM. These API calls can be accessed through a common entry that can be obtained through interrupt 1Ah. This PXE API test program attempts to verify all the API calls for their existence. It also verifies the contents of all the registers modified by the call. The test results are stored in a log file on the client and on the proxyDHCP server. Tthe APIs are tested in the following sequence: GET_CACHED_INFO, UDP, TFTP and UNDI. Since the PXE API test program shuts down the connection between the UNDI layer and the UDP/IP protocol stack in the boot ROM, the TFTP and UDP APIs will not be available if the PXE API test program is run for the second time.

5.1.4 Test Analysis

The Bootserver analyzes the results of the three tests carried out by the clients. This is done only when “Packet Analysis Test On” is enabled.

The Packet Analysis screen provides options to view and control. (It is necessary to restart the PXE Service after switching Packet Analysis on/off for the changes to take affect.)



Waiting Thread Count displays the number of clients that are running the API Test which the Bootserver is waiting to analyze. It is possible for the Bootserver to become confused about the number of running clients (if the clients hang during the test). The Stop button can be used to manually stop all client test monitoring. It should be used only when necessary.

The Log File Size Limit provides control over the size of the circular buffer log files into which packets are captured (Dhcppkt.txt and Testsum.txt files in the <mac-addr> directory).

The Delete Passed Test Directory check box provides a way to save disk space by deleting directories for those clients that have passed the test.

NOTE:

For test analysis, it is recommended that two server machines be setup. Make the first machine a standalone DHCP server. On the second machine install the proxyDHCP and Bootserver. This makes the analysis more thorough as the packets captured by the proxyDHCP are also tested.

The outcome of this analysis is displayed on the client screen, thus reducing end user effort.

5.2 Test Logs

During the API Test a RAMDrive is created on the client machine with 8MB of extended memory. All client test output files will be created on this virtual A: drive. In addition, if the universal NDIS driver successfully loads, the <install directory>\TestLOG share of the Bootserver is mapped to a drive by the client and the test log files are copied into the client's <mac-addr> subdirectory.

The proxyDHCP service on the test server will place the results of the Packet Analysis test in log files in the <install directory>\TestLOG client subdirectory. The test logs are cumulative. Therefore, the client's subdirectory should be deleted before performing a new test boot.

In summary:

- The test logs are located in the subdirectory: <install directory>\Testlog.
- The log files for a particular client machine are in the subdirectory with the last eight characters of the client's MAC address.

This client's <mac-addr> subdirectory will contain these files:

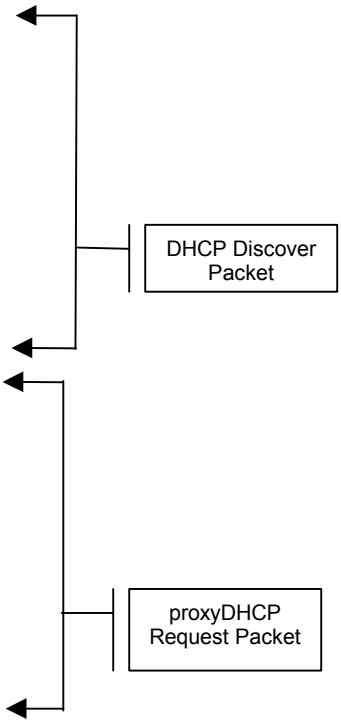
File	Description	
TESTSUM.TXT	Contains a summary of the results of the PXE API test and the Packet Analysis tests. The packet analysis summary is first, followed by the summary results of the PXE API test.	
DHCPDISC.TXT	1. The client's DHCPDISCOVER packet	For more information refer to the PXE 2.0 specification: Preboot API Service Descriptions - GET CACHED INFO
DHCP_ACK.TXT	2. The DHCP server's DHCPACK packet	
BOOT_ACK.TXT	3. The Boot Server's Discover Reply packet, which contains Option #60 set to "PXEClient", a valid bootfile name, and may contain MTFTP options	
NDISTEST.TXT	Contains the detailed output from performing a series of file transfers to stress test the NDIS interface	
DHCPPKT.TXT	Contains a hex dump of the DHCPDiscover or CachedReply packet and a detailed analysis of the PXE options found or missing. DHCPPKT.TXT may contain multiple packet analysis depending on how many DHCP transactions were performed.	

The following are example contents of the TESTSUM.TXT file. This example illustrates the results of a client that sends a DHCP Discover when it first boots and then follows up with a proxyDHCP Request. This client has failed the compatibility test because it does not include a UUID in the DHCP or cached packets. (The third packet analysis is from the DOS TCP/IP stack that is loaded during the API Test. This was a DHCP transaction that occurred after the client booted.)

The text after the packet analysis indicates the client passed all of the APITests.

testlog/<mac-addr>/TESTSUM.TXT:

```
================================================================================
12/19/97 13:06:20 : Results of Packet Analysis For Client:C90C945A
================================================================================

        Invalid or missing Client Machine Identifier(UUID) (97,0x61)

FAILED: Packet does not comply with PXE protocol as specified in
        PXE v 2.0 due to the errors listed above.

--------------------------------------------------------------------------------
                END OF ANALYSIS
--------------------------------------------------------------------------------

********************************************************************************


================================================================================
12/19/97 13:06:20 : Results of Packet Analysis For Client:C90C945A
================================================================================

        Invalid or missing Client Machine Identifier(UUID) (97,0x61)

FAILED: Packet does not comply with PXE protocol as specified in
        PXE v 2.0 due to the errors listed above.

--------------------------------------------------------------------------------
                END OF ANALYSIS
--------------------------------------------------------------------------------

********************************************************************************
```



```
=====================================================================
12/19/97 13:07:01 : Results of Packet Analysis For Client:C90C945A
=====================================================================

       Invalid or missing Vendor Class Identifier (60,0x3C)
       Invalid or missing Client System Architecture (93,0x5C)
       Invalid or missing Client Network Interface Identifier (94,0x5D)
       Invalid or missing Client Machine Identifier(UUID) (97,0x61)

FAILED: Packet does not comply with PXE protocol as specified in
       PXE v 2.0 due to the errors listed above.

---------------------------------------------------------------------
               END OF ANALYSIS
---------------------------------------------------------------------

*********************************************************************
```

DHCP Discover from DOS TCP/IP during APITest

Note that this is not a packet generated by PXE, but rather one that occurred after booting.

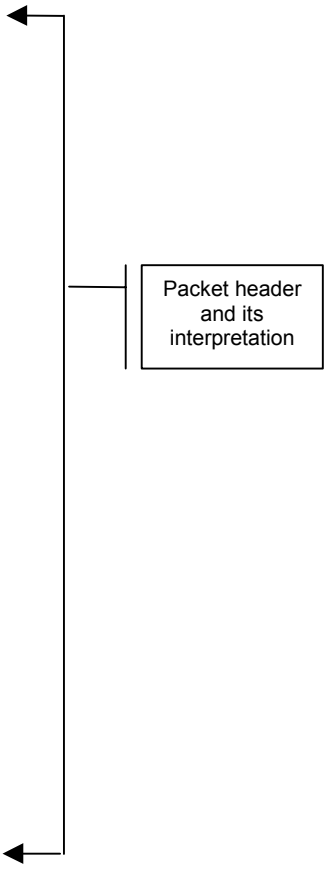
```
GET_DHCP_DISCOVER: Passed..
GET_DHCP_ACK: Passed..
GET_CACHED_REPLY: Passed..
UNDI_GET_INFO: Passed..
GET_DHCP_DISCOVER: Passed..
GET_DHCP_ACK: Passed..
GET_CACHED_REPLY: Passed..
UDP_OPEN: Passed..
UDP_WRITE: Passed..
UDP_READ: Passed..
UDP_CLOSE: Passed..
TFTP_GET_FILE_SIZE: Passed..
TFTP_OPEN: Passed..
TFTP_READ: Passed..
TFTP_CLOSE: Passed..
TFTP_READ_FILE: Passed..
UNDI_SHUTDOWN: Passed..
UNDI_INIT: Passed..
UNDI_OPEN: Passed..
UNDI_TRANSMIT: Passed..
UNDI_GET_MCAST_ADDR: Passed..
UNDI_SET_MCAST_ADDR: Passed..
UNDI_SET_STATION_ADDR: Passed..
UNDI_SET_PACKET_FILTER: Passed..
UNDI_GET_NIC_TYPE: Passed..
UNDI_GET_IFACE_INFO: Passed..
UNDI_GET_INFO: Passed..
UNDI_GET_STAT: Passed..
UNDI_INITIATE_DIAGS: Call is present but not supported.
               This is an optional call.
UNDI_FORCE_INTERRUPT: Passed..
UNDI_GET_MCAST_ADDR: Passed..
UNDI_GET_NIC_TYPE: Passed..
UNDI_GET_IFACE_INFO: Passed..
UNDI_CLOSE: Passed..
UNDI_RESET_NIC: Passed..
UNDI_SHUTDOWN: Passed..
```

Results of API Testing

The server log file (Dhcppkt.txt) and the client log files (DhcpDisc.txt, Dhcp_Ack.txt and Boot_Ack.txt) present under the <mac-addr> directory on server have detailed interpretation of the packets. The following are partial samples from some of those files:

Testlog/<mac-addr>/DHCPPKT.TXT:

```
======================================================================
08/25/99 15:58:12 : Start of Packet Analysis For Client:00AA00D10B17
======================================================================
----------------------------------------------------------------------
Fixed portion of DHCP Packet sent by client:00AA00D10B17
----------------------------------------------------------------------
     0000  01 01 06 00 01 D1 0B 17 04 00 80 00 00 00 00 00  .....Ñ....□.....
     0010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 AA 00 D1  ..............ª.Ñ
     0020  0B 17 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     00a0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     00b0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     00c0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     00d0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     00e0  00 00 00 00 00 00 00 00 00 00 00 00              ............

Op Code                     (op)          1 (0x01) BOOTREQUEST
Hardware Type               (htype)       1 (0x01) 10Mb Ethernet
Hardware Address Length     (hlen)        6 (0x06)
Hops                        (hops)        0 (0x00)
Transaction ID              (xid)         386650369 (0x170bd101)
Seconds                     (secs)        4 (0x0004)
Flags                       (flags)       128 (0x0080)
Client IP Address           (ciaddr)      0.0.0.0
Your   IP Address           (yiaddr)      0.0.0.0
Server IP Address           (siaddr)      0.0.0.0
Relay  IP Address           (giaddr)      0.0.0.0
Client Ethernet Address     (chaddr)      00-aa-00-d1-0b-17
Server Host Name            (sname)       <blank>
Boot File Name              (file)        <blank>
```



```
------------------------------------------------------------------------
DHCP Options sent by client:00AA00D10B17
------------------------------------------------------------------------
     0000  63 82 53 63 35 01 01 37 0D 01 03 3C 2B 43 80 81  c,Sc5..7...<+C□□
     0010  82 83 84 85 86 87 39 02 04 EC 61 11 00 14 C3 34  ‚ƒ„…†‡9..ìa...Ã4
     0020  C9 1A F5 11 D2 8D D3 00 A0 C9 68 8B A9 5D 02 00  É.õ.Ò□Ó. Éh‹©]..
     0030  00 5E 03 01 02 01 3C 20 50 58 45 43 6C 69 65 6E  .^....< PXEClien
     0040  74 3A 41 72 63 68 3A 30 30 30 30 30 3A 55 4E 44  t:Arch:00000:UND
     0050  49 3A 30 30 32 30 30 31 FF 00 00 00 00 00 00 00  I:002001ÿ.......
     0060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     00a0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     00b0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     00c0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     00d0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     00e0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     00f0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0100  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0120  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
     0130  00 00 00 00 00 00 00 00                          ........


Magic Cookie   [OK]
Option  53(0x35) [DHCP Packet Type = DHCPDISCOVER]
Option  55(0x37) [Parameter Request List = Subnet Mask-1,
          Router Option-3, Vendor Class Identifier-60, Vendor Specific Info-43,
          Boot File Name-67, Vendor NBP-128, Vendor NBP-129,
          Vendor NBP-130, Vendor NBP-131, Vendor NBP-132,
          Vendor NBP-133, Vendor NBP-134, Vendor NBP-135, ]
Option  57(0x39) [Max DHCP Message Size = 1260 octects]
Option  97(0x61) [GUID = {Type = 00} 14C334C91AF511D28DD300A0C9688BA9]
Option  93(0x5d) [System Architecture = (0x0000) X86PC]
Option  94(0x5e) [Network Device Interface = UNDI Version 2.1]
Option  60(0x3c) [Vendor Class Identifier = PXEClient:Arch:00000:UNDI:002001]


------------------------------------------------------------------------
               PACKET ANALYSIS
------------------------------------------------------------------------


SUCCESS: Packet complies with PXE protocol as specified in
       PXE Specification  Version 2.0.

------------------------------------------------------------------------
               END OF ANALYSIS
------------------------------------------------------------------------

************************************************************************
```

Packet options and its interpretation

Testlog/<mac-addr>/DHCPDISC.TXT:

```
Op Code                     (op)        1 (0x01) BOOTREQUEST
Hardware Type               (htype)     1 (0x01) 10Mb Ethernet
Hardware Address Length     (hlen)      6 (0x06)
Hops                        (hops)      0 (0x00)
Transaction ID              (xid)       386650369 (0x170bd101)
Seconds                     (secs)      4 (0x0004)
Flags                       (flags)     128 (0x0080)
Client IP Address           (ciaddr)    0.0.0.0
Your   IP Address           (yiaddr)    0.0.0.0
Server IP Address           (siaddr)    0.0.0.0
Relay  IP Address           (giaddr)    0.0.0.0
Client Ethernet Address     (chaddr)    00-aa-00-d1-0b-17
Server Host Name            (sname)     <blank>
Boot File Name              (file)      <blank>

Options:
Magic Cookie   [OK]
Option  53(0x35) [DHCP Packet Type = DHCPDISCOVER]
Option  55(0x37) [Parameter Request List = Subnet Mask-1,
          Router Option-3, Vendor Class Identifier-60, Vendor Specific Info-43,
          Boot File Name-67, Vendor NBP-128, Vendor NBP-129,
          Vendor NBP-130, Vendor NBP-131, Vendor NBP-132,
          Vendor NBP-133, Vendor NBP-134, Vendor NBP-135, ]
Option  57(0x39) [Renewal (T1) Time Value = 1260 octects]
Option  97(0x61) [GUID = {Type = 00} 14C334C91AF511D28DD300A0C9688BA9]
Option  93(0x5d) [System Architecture = (0x0000) X86PC]
Option  94(0x5e) [Network Device Interface = UNDI Version 2.1]
Option  60(0x3c) [Vendor Class Identifier = PXEClient:Arch:00000:UNDI:002001]
```

Full packet interpretation

```
0000: 01 01 06 00 01 d1 0b 17 04 00 80 00 00 00 00 00   ................
0010: 00 00 00 00 00 00 00 00 00 00 00 00 aa 00 d1      ................
0020: 0b 17 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00d0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00e0: 00 00 00 00 00 00 00 00 00 00 00 00 63 82 53 63   ............c.Sc
00f0: 35 01 01 37 0d 01 03 3c 2b 43 80 81 82 83 84 85   5..7...<+C......
0100: 86 87 39 02 04 ec 61 11 00 14 c3 34 c9 1a f5 11   ..9...a....4....
0110: d2 8d d3 00 a0 c9 68 8b a9 5d 02 00 00 5e 03 01   ......h..]...^..
0120: 02 01 3c 20 50 58 45 43 6c 69 65 6e 74 3a 41 72   ..< PXEClient:Ar
0130: 63 68 3a 30 30 30 30 3a 55 4e 44 49 3a 30 30   ch:00000:UNDI:00
0140: 32 30 30 31 ff 00 00 00 00 00 00 00 00 00 00 00   2001............
```

Also, a common log file, Testsum.txt, gets created under the “testlog” directory. This file has the status of all API Tests that the server analyzed. The following is a sample of its contents:

testlog/TESTSUM.TXT:

Date	Time	MAC Address	GUID	Result
08/20/99	16:50:35	00-AA-00-D1-0B-17	0014C334-C91A-F511-D28D-D300A0C9688B	PASS
08/20/99	16:52:28	00-90-27-5B-AE-0F	F3CC16FE-FC69-11D2-AE45-08000918F676	PASS
08/20/99	16:54:59	00-90-27-50-B0-0D	557B8EEA-FC80-11D2-AE45-08000918F676	STOPPED
08/20/99	16:57:28	00-90-27-50-AC-D9	F3CC16F1-FC69-11D2-AE45-08000918F676	STOPPED
08/20/99	16:54:59	00-90-27-50-B0-0D	557B8EEA-FC80-11D2-AE45-08000918F676	PASS
08/23/99	17:12:06	00-AA-00-D1-0E-32	0014C337-C91A-F511-D28D-D300A0C9688B	FAIL
08/20/99	16:57:28	00-90-27-50-AC-D9	F3CC16F1-FC69-11D2-AE45-08000918F676	PASS
08/23/99	17:34:51	00-AA-00-D1-0B-17	0014C334-C91A-F511-D28D-D300A0C9688B	PASS
08/25/99	15:56:24	00-AA-00-D1-0B-17	0014C334-C91A-F511-D28D-D300A0C9688B	PASS

If Packet Analysis is enabled on the server, the client waits for the Bootserver to analyze the test when the client finishes performing API Test. The results of analysis are displayed on the client screen. A sample of the screen follows:

```
Waiting for Server Analysis...

Packet Analysis  :  PASS

Api Test         :  PASS

NDIS Test        :  PASS

=====================================

Result           :  PASS

=====================================
C:\RESULTS>
```

The final directory on client when the test ends (“Results”) contains essential files that can be analyzed for reasons of failure. This directory is created on the ramdrive.

6. Release Notes

ROMs, NBPs and services are affected by this release.

6.1 PXE Boot ROM

PXE PDK Release Date	*PXE PDK Ver #*	*Wired for Management Baseline Version*
6/21/2000	**V3.0 bld 082**	**V2.0**

Boot ROM - PXE - 2.0 (Build 082) Code Changes

ROM Init

- If the PXE configuration word (3Bh) in the EEPROM was not configured, the ROM was reported to hang. If the EEPROM is not configured, the ROM initialization code now defaults to BBS bootstrap and use built-in BaseCode, if present.

Runtime Loader

- The expansion ROM BAR in the PCI configuration space is no longer used to locate the PXE runtime image, only the FLASH BAR (BAR 2) is used.
- All ROM images included in this PDK now display an unsupported development version message when booting. The ROM images included in this PDK cannot be used in released product as described in the PDK license.
- Exit/Status codes returned from Start BaseCode API are no longer checked. When control is returned the Stop BaseCode API is issued. If a keep code or an error is returned, the BaseCode will not be unloaded. This fixes a bug where the PXE split ROM would hang when returning to the system BIOS if there were no DHCP or proxyDHCP servers.

BaseCode

- The vendor-information (option 43) was being ignored if the class-identifier (option 60) was either not present or came after the vendor-information. The BaseCode now accepts any vendor-information that has any valid PXE vendor options, whether or not there is a class-identifier.
- The requested DHCP options in the DHCP discover packet were incorrect. The router option (tag 3) was requested twice and the time offset option (tag 2) was not requested. This has been fixed.
- The timer in the NIC shutdown code was replaced due to reports of hanging during shutdown. The new code reads the 8254 timer registers instead of the tick counter in the BIOS data segment.

Early UNDI

- No changes.

UNDI

- Interrupts are now disabled when reading the 8254 timer. No bugs were reported against this, it is just cautionary.

4/27/2000	V3.0 bld 079	V2.0

Boot ROM - PXE - 2.0 (Build 079) Code Changes

ROM Init

- Removed all ROM initialization messages and bootstrap selection code. These operations are now done using a separate DOS command line utility, EEPROM.EXE, that is included in this PDK.

Runtime Loader

- Scan for split BaseCode ROM could hang when checking for BaseCode ROM signature. The check address is now validated before checking the signature.

BaseCode

- The last TFTP Ack packet was being dropped, sometimes, on faster client machines. An additional 55 millisecond delay was added before the NIC is shutdown to be sure that the last packet gets written to the network.
- The seconds field of the DHCP packet was not being stored in network order. This was not causing any known issues, it was just not following the DHCP RFC.
- The seconds field being sent in the DHCP Discover packet was being changed when sent in the DHCP Request packet. This was not causing any known issues, it was just not following the DHCP RFC.
- A DHCP Decline packet is now being sent if the client cannot get a response from the proxyDHCP service on port 4011. This happens when a DHCP Offer packet is sent with 'PXEClient' and the proxyDHCP service is not installed or not working properly. Also, the client now displays the message "PXE-E55: proxyDHCP service did not reply to request on port 4011."
- The size of the PXE client ARP cache has been increased from 2 entries, to 4. This is done so the client will not ARP every packet when the DHCP, proxyDHCP and TFTP services are all on different servers.
- The TFTP source port number would change to the last UDP write source port number + 1. This has been fixed.
- If the client receives proxyDHCP offers and does not receive DHCP or BOOTP offers it now displays this message: PXE-E52: proxyDHCP offers were received. No DHCP offers were received.
- Top half of EAX was being trashed during NIC interrupts. All registers are now saved & restored during interrupt handling.
- UDP & TFTP APIs now ignore the gateway IP address parameter if the station, gateway and destination/server IP addresses are all on the same subnet.
- The client no longer does a TFTP get file size if the file size option is returned in the bootserver reply.
- The client now uses the BIOS tick counter instead of the hardware timer to count seconds. This is done because some clients are using different hardware timer speeds.

Early UNDI

- No changes.

UNDI

- Top half of EAX was not being preserved during UNDI API calls. All registers are now saved & restored. API call status is returned in AX only.

1/21/2000	V3.0 bld 078	V2.0

Boot ROM - PXE - 2.0 (Build 078) Code Changes

ROM Init

- Support for spoofing Int 15h, AX=E801h (Get memory size for >64M configurations) service has been removed. This service does not need to be spoofed because it does not report memory below 1M which is where the PXE runtime image is located. The spoofing code in earlier PXE ROM versions was not implemented correctly and caused some Linux implementations to only recognize the first 24M of memory.

Runtime Loader

- No changes.

BaseCode

- The seconds field in the DHCP packet was not being byte swapped.
- TFTP Read File was returning PXENV_STATUS_FAILURE for all TFTP errors that occurred. TFTP Read File has been changed to return TFTP error status codes to the calling program.
- Changed menu selection bar from direct video memory writes to video BIOS write string calls (Int 10h, AH=13h). This was done so the menu selection bar could be redirected to COM: ports.
- Added code to parse the client IP, BOOTP server IP and subnet mask when a BOOTP server is used.
- Removed code that checked for the DHCP message type (option 53) in the unicast proxyDHCP reply. The ROM was only accepting packets with a message type of DHCPACK (5).
- Added minimum length check to BIS SMBIOS structure check.
- SIADDR field in DHCP/BOOTP packet is used for TFTP server IP address if it is not 0.0.0.0.

Early UNDI

- No changes.

UNDI

- Changed the UNDI state check at the beginning of the Set Station Address API so that it would only operate when the UNDI was initialized and fail when the UNDI was opened.

12/1/99	V3.0 bld 077	V2.0

Boot ROM - PXE - 2.0 (Build 077) Code Changes

ROM Init

- First time initialization (power on out of the box) of a PXE NIC was not setting the default bootstrap type correctly in SMBIOS v2.3 systems. In systems with BIOSes with BBS support reported in the BIOS information (Type 0) table that did not also support the BBS runtime functions, the PXE NIC ROM would set the default bootstrap to Int 18h instead of BBS. This has been fixed. Also, if SMBIOS v2.3 or later is supported the BBS runtime functions are no longer checked by the ROM. In these BIOSes the BBS support flag must be implemented in the BIOS information (Type 0) table.
- 4 second delay to keep initialization error messages on the screen long enough to read was always being run in LOM images. Speeds up LOM image boots 4 seconds.

Runtime Loader

- If a monolithic PXE ROM was set to use the BaseCode built into the BIOS and the BIOS did not have a built in BaseCode, error message "PXE-EC1: BaseCode not found" was being displayed. This message is no longer displayed in monolithic ROMs because they always have a BaseCode.

BaseCode

- No changes.

Early UNDI

- No changes.

UNDI

- No changes.

10/29/99	V3.0 bld 074	V2.0

Boot ROM - PXE - 2.0 (Build 074) Code Changes

ROM Init

- There are now four LOM images (e100_m.bbs, e100_s.bbs, e100_m.i18 and e100_s.i18). The old LOM images (e100_m.lom and e100_s.lom) have been removed. Which LOM image to use depends on the system BIOS. If BBS is supported use the e100_?.bbs images. If BBS is not supported, use the e100_?.i18 images.
- Initialization menu and delays have been removed from the PXE LOM images.

Runtime Loader

- No changes.

BaseCode

- DMI BIOSes w/o GUIDs were displaying a bogus GUID on the screen and sending a bogus GUID in the DHCP discover packet. GUIDs are now properly detected in all DMI and SM BIOSes.
- DHCP code would retry DHCP discover if the first DHCP offer packets received did not have a class identifier of 'PXEClient'. Now any DHCP offer that has 'PXEClient' or a bootfile will be accepted.
- Clear screen has been removed from remote boot prompt/menu display. The screen used to be cleared if the menu reached the bottom of the screen.
- Address of !PXE structure is now placed on the CPU stack, as defined by the PXE 2.1 specification. It was being placed in the DX:AX (seg:off) register pair.
- **Early UNDI**
- No changes.

UNDI

- PXENV_STOP_UNDI now works only if PXENV_START_UNDI has been called. PXENV_STOP_UNDI was not checking the state of the UNDI and was always returning success, even if the UNDI was not started. If PXENV_STOP_UNDI was called before PXENV_START_UNDI, the Int 1Ah vector would have been corrupted.

8/27/99	V3.0 bld 072	V2.0

Boot ROM - PXE - 2.0 (Build 072) Code Changes

ROM Init

- PCI data structure revision changed from 0x01 to 0x00 to match PCI specification.

Runtime Loader

- Size of free base memory reported in 40:13h is used to determine if there is enough free base memory. ROM no longer scans for a block of zero filled base memory.

BaseCode

- Address of !PXE structure is passed on the stack to NBP, instead of address of PXENV+ structure, to match PXE specification.
- Client and DHCP IP addresses are displayed when client does not get bootfile or PXE discovery tags.
- If client has a GUID, even an invalid GUID of all zeroes or ones, the GUID is displayed with the client MAC address.
- PXENV_UNLOAD_STACK now unhooks the IRQ being used by the BaseCode.
- "PXE-E7A: Client could not locate a secure boot server" is displayed if client does not get security information when BIS is enabled.
- "PXE-E7B: Missing MTFTP server IP address" is displayed if there is no PXE discovery information and the DHCP SIADDR field is 0.0.0.0.
- The client network interface identifier tag (DHCP option 94) and the UNDI version stored in the class identifier tag (option 60) "PXEClient:Arch:00000:UNDI:002001" were hard coded. These are now read from the version field in the UNDI ROM ID structure.
- UNDIclose was not being called after an error is returned by TFTPopen, TFTPreadfile, TFTPgetfilesize or UDPopen. This caused all calls to any of these APIs to fail after the first error.
- The DHCP SIADDR field is used as the MTFTP server IP address if there is no discovery information available in the DHCP packet and there is no Proxy packet. If the SIADDR field is 0.0.0.0, the DHCP server IP address is used.
- **Early UNDI**
- No changes.

UNDI

- When PXENV_UNDI_TRANSMIT returned a status code of OUT_OF_RESOURCES, the exit code is now PXENV_EXIT_FAILURE, instead of PXENV_EXIT_SUCCESS.
- When a BaseCode API is issued after BaseCode is unloaded, a status/exit code pair of PXENV_STATUS_UNDI_INVALID_FUNCTION/PXENV_EXIT_FAILURE is returned, instead of PXENV_STATUS_SUCCESS/PXENV_STATUS_UNDI_INVALID_FUNCTION.

6/28/99	V3.0 bld 071	V2.0

Boot ROM - PXE - 2.0 (Build 071) Code Changes

ROM Init

- Branch past code to clear initialization messages when initialization messages are not being displayed. Fixes flashing/clearing screen seen in some systems when booting.

Runtime Loader

- The LOM version of PXE was shrinking the ROM image in upper memory when POST Memory Manager is not present in the BIOS. This caused the ROM to fail with a PXE-EC8 message on boot. This has been fixed.

BaseCode

- DHCP IP address being displayed during the remote boot was not correct when booting through Linux software routers. The address being displayed was the router IP address and not the DHCP server IP address. This has been corrected.
- The TFTP APIs will exit early (not retry) if an error response is received from the TFTP server. The client used to retry, and get the same error again and again, until the retry timeout occurred.
- TFTP error messages sent by the TFTP server are no longer displayed when the TFTP APIs are used by NBPs. The NBPs are now responsible for displaying their own TFTP error messages.
- Possible NULL pointer usage removed in DHCP option parsing code. "if (o->ovrload & 0x??) …" statements changed to "if (o->ovrload && (o->ovrload & 0x??) …".
- UDP read failed was re-assembling fragmented packets. The checksum in the first packet was being corrupted by the following fragments. This has been fixed.
- **Early UNDI**
- The LOM early UNDI loader would not work unless the option ROM was initialized by the BIOS, the call to the loader would return with the error code: 0xC8 (The !PXE structure could not be found). This has been fixed.

UNDI

- No changes.

4/2/99	V3.0 bld 068	V2.0

Boot ROM - PXE - 2.0 (Build 068) Code Changes

ROM Init

- Added code to increase exit timeout when an error message is being displayed from 2-3 seconds to 5-6 seconds so initialization error messages can be read.
- Only accept 'Y' or 'N' to the "display initialization messages" question in the bootstrap selection menu. Before this any key except 'Y' was treated as an 'N'.
- Relocated code that was clearing the "Hold both shift keys" message so it only runs when initialization messages are displayed. This is so the ROM will not display any characters during option ROM initialization (scan) unless initialization messages are displayed.

Runtime Loader

- No changes.

BaseCode

- Added code to delay 5-6 seconds when an error message is displayed before returning control to the BIOS so runtime error messages can be read.

UNDI

- No changes.

2/8/99	V3.0 bld 067	V2.0

Boot ROM - PXE - 2.0 (Build 067) Code Changes

- A 'Break' statement was deleted from the DHCP module. This caused all DHCP/BOOTP packets without 'PXEClient' but with a bootfile to be ignored by the client.
- Removed last six references to PXENV+ structure. The API entry point routines were taking the UNDI data segment and BaseCode code segment from the PXENV+ structure. These are now coming from the !PXE structure.
- Removed 'beta' designator from version number.

6.2 PXE NT Services

PXE PDK Release Date	*PXE PDK Ver #*	*Wired for Management Baseline Version*
6/21/2000	**V3.0 bld 082**	**V2.0**
NT Services - PXE-2.0 (Build 082) Code Changes		
• No changes.		
4/27/2000	**V3.0 bld 079**	**V2.0**
NT Services - PXE-2.0 (Build 079) Code Changes		
• Exclusion scoping has been added • PxeMtftp ignores duplicate acks; retransmits data on timeout. This is for both – tftp and mtftp • It is now possible to stop just the TFTP and use a third party software. • Bootserver returns file size option (13). • Api test file names changes: cpkt1.txt is dhcpdisc.txt, cpkt2.txt is dhcp_ack.txt and cpkt3.txt is boot_ack.txt. • New helper utility, PxeReg60, automates setting of option 60 for Win 2000 MS Dhcp Server.		
1/21/2000	**V3.0 bld 078**	**V2.0**
NT Services - PXE-2.0 (Build 078) Code Changes		
• PXEClientTester.dll - Improved API test logs.		
12/1/99	**V3.0 bld 077**	**V2.0**
NT Services - PXE-2.0 (Build 077) Code Changes		
• No changes.		
10/29/99	**V3.0 bld 074**	**V2.0**
NT Services - PXE-2.0 (Build 074) Code Changes		
• Scoping has been added to filter clients based on several criteria. • While doing packet analysis, boot server probes the clients running API Test. It generates a final result that gets displayed at the client terminal.		
8/27/99	**V3.0 bld 072**	**V2.0**
NT Services - PXE-2.0 (Build 072) Code Changes		
• No changes.		
6/28/99	**V3.0 bld 071**	**V2.0**
NT Services - PXE-2.0 (Build 071) Code Changes		
• No changes.		
4/2/99	**V3.0 bld 068**	**V2.0**
NT Services - PXE-2.0 (Build 068) Code Changes		
• Text output in the PXE client test module (pxeclienttester.dll) has been updated. References to PXE-1.0 have been changed to PXE-2.0.		
2/8/99	**V3.0 bld 067**	**V2.0**
NT Services - PXE-2.0 (Build 067) Code Changes		
• No changes.		

6.3 PXE NBPs & Image Contents

PXE PDK Release Date	*PXE PDK Ver #*	*Wired for Management Baseline Version*
5/26/2000	**V3.0 bld 082**	**V2.0**

NBP & Images - PXE - 2.0 (Build 082) Code Changes

BStrap.0 NBP
- No changes.

APITest.0 NBP
- No changes.

DOSUNDI.0 NBP
- No changes.

RIStrap.0 NBP
- New to this PDK.

API Test
- No changes.

NDIS Driver
- All 32-bit registers are now preserved in the protocol driver and interrupt service routines. The top half of the 32-bit general registers (EAX, EBX, ECX, EDX) were not being preserved. This did not seem to cause any problems in the Microsoft DOS network client stack but was causing problems in the IBM DOS network client stack.

4/27/2000	**V3.0 bld 079**	**V2.0**

NBP & Images - PXE - 2.0 (Build 079) Code Changes

BStrap.0 NBP
- Do not set gateway IP address in TFTP get file size and TFTP read file API calls if server is on same subnet as client.
- Do not issue TFTP get file size API call if DHCP boot file size option is present and valid.

APITest.0 NBP
- API test file names changes: cpkt1.txt is dhcpdisc.txt, cpkt2.txt is dhcp_ack.txt and cpkt3.txt is boot_ack.txt.
- API test scans memory for PXE structure and compares it with one obtained using INT 1Ah.
- TFTP get file size and TFTP read file API calls will not use gateway IP address if station and server IP addresses are on the same subnet.

DOSUNDI.0 NBP
- TFTP get file size and TFTP read file API calls will not use gateway IP address if station and server IP addresses are on the same subnet.

API Test
- No changes.

NDIS Driver
- No changes.

1/21/2000	**V3.0 bld 078**	**V2.0**

NBP & Images - PXE - 2.0 (Build 078) Code Changes

BStrap.0 NBP
- Menu selection bar is now displayed using video BIOS write string calls (Int 10h, AH=13h) instead of direct video memory writes.

APITest.0 NBP
- No changes.

DOSUNDI.0 NBP
- No changes.

API Test
- "UNDI Set Station Address" API test is now run when the UNDI is initialized, not when the UNDI is opened.
- Improved API test logs.

NDIS Driver
- No changes.

12/1/99	V3.0 bld 077	V2.0
NBP & Images - PXE - 2.0 (Build 077) Code Changes		
APITest.0 NBP • No changes. **DOSUNDI.0 NBP** • No changes. **API Test** • No changes. **NDIS Driver** • No changes.		
10/29/99	**V3.0 bld 074**	**V2.0**
NBP & Images - PXE - 2.0 (Build 074) No changes		
APITest.0 NBP • No changes. **DOSUNDI.0 NBP** • No changes. **API Test** • At the start, the test determines if the PXE Boot server is doing packet analysis. If packet analysis is being done, then the final result computed by the boot server is displayed on the client terminal. • BINL_INFO references changed to CACHED_INFO to match PXE documentation. **NDIS Driver** • No changes.		
8/27/99	**V3.0 bld 072**	**V2.0**
NBP & Images - PXE - 2.0 (Build 072) Code Changes		
APITest.0 NBP • Will now work with old universal NDIS drivers (v1.0). WfM-1.1a entry point of offset zero in the BaseCode code segment is preserved after the RAMdisk image is downloaded. **DOSUNDI.0 NBP** • Will now work with old universal NDIS drivers (v1.0). WfM-1.1a entry point of offset zero in the BaseCode code segment is preserved after the RAMdisk image is downloaded. **API Test** • No changes. **NDIS Driver** • No changes.		
6/28/99	**V3.0 bld 071**	**V2.0**
NBP & Images - PXE - 2.0 (Build 071) Code Changes		
Changed from InstallShield 3.1 to InstallShield 5.5 **APITest.0 NBP** • No changes. **DOSUNDI.0 NBP** • No changes. **API Test** • No changes. **NDIS Driver** • No changes.		

4/2/99	V3.0 bld 068	V2.0

NBP & Images - PXE - 2.0 (Build 068) Code Changes

NBP

- The check for illegal server IP addresses in the PXE Boot Server list (PXE tag 8) was not byte swapping the server IP addresses when checking for multicast addresses. This caused IP addresses x.x.x.240 through x.x.x.245 to be considered invalid. This has been fixed.

API Test

- The API test program was filling the gateway IP address with *(UINT32 far *)0 instead of (UINT32)0 when doing the UDP write test. This did not cause the test to fail, because the packet would go to the network, but the contents of the packet were not correct.
- API test program was displaying “Invalid function” for some of the API tests being done. This message could be confusing and has been changed to “Passed”.
- API test program was identifying PXENV_UNDI_GET_IFACE_INFO as an optional call. This was true for WfM-1.1a, but is no longer true for WfM-2.0. This has been corrected.

NDIS Driver

- Found where a receive interrupt could have been lost when returning from handling another receive interrupt. This could have been responsible for NDIS.DOS hanging occasionally during large transfers or on busy networks.

7. Third-Party Design Support

The following independent SW vendors (ISV) are both available to provide design support for PXE-compliant boot ROMs.

3COM/Lanworks Technologies Co.
2425 Skymark Ave.
Mississauga, ON
CANADA L4W 4Y6
Contact: Mark Kuess
Phone: (905) 238-5528 ext. 166
Fax: (905) 238-9407
Email: mark_kuess@3com.com
Internet: http://www.3com.com/managedpc

bootix Technology GmbH
Geranienstr. 19
D-41466 Neuss Germany
Phone: + 49 2131 7486-0
Fax: + 49 2131 7486-26
Email: info@bootix.com or info@boomanage.com
Internet: http://www.bootix.com or http://www.bootmanage.com

Elisa Research Inc.
4450 Enterprise Street, Ste #106
Fremont, CA 94538
Phone: (510) 770-4920
Fax: (510) 770-4930
Email: pwang@elisaresearch.com
Internet: http://www.elisaresearch.com/

Insyde Software, Inc. (for mobile PXE development assistance)
One Innovation Drive
Natick, MA 01760
Fax: (508) 647-5546
Email: getinfo@insydesw.com

8. PXE 2.0 Operation and Troubleshooting FAQs

8.1 Known Issues

8.1.1 Early UNDI cannot be used in big real mode

It is possible for the system BIOS to load the UNDI out of a PXE option ROM before the end of P.O.S.T. and use the UNDI APIs for remote management/configuration. The BIOS developers must not make UNDI API calls when the processor is in big real mode. If UNDI API calls are made in big real mode the UNDI API call may hang or may trash other system memory.

8.1.2 Client ARPs every TFTP packet when DHCP, proxyDHCP and TFTP are on three different machines

This happens in builds 067 through 078. Starting with build 079 the size of the client ARP cache has been increased.

8.1.3 The last TFTP Ack packet from the client is dropped

This can happen with builds 067 through 078 when being run on faster (> 400MHz) machines or on busy networks. Starting with build 079, an additional 55ms timeout has been added when shutting down the NIC.

8.1.4 Top half of EAX trashed by UNDI API calls and BaseCode interrupt handler

There is no work-around for this issue. All clients should be updated to at least build 079.

8.1.5 Top half of 32-bit general registers trashed by universal NDIS driver routines

All universal NDIS versions before 082 did not save EAX, EBX, ECX or EDX when called by the protocol driver. This did not seem to cause any problems with the Microsoft DOS network stack but some odd behaviours were observed with the IBM DOS network stack. All vendors using the Intel universal NDIS driver should upgrade to build 082 or later.

8.1.6 PXE ROMs will not accept DHCP offer with option 43 before option 60

All Intel PXE ROMs before build 082 would not see the vendor-encapsulated-options (option 43) if it was placed before the vendor-client-identifier (option 60). Either change the DHCP server settings so option 43 is listed after option 60, or update the client firmware to build 082 or later.

8.2 FAQ w/ A

Q1. Why are there three different versions of the PXE boot ROMs for the Intel 82557/82558/82559-based network controllers?

A1. E100_M.NIC is designed to run from the boot PROM of a NIC and does not require PXE BaseCode to be installed in the BIOS. This code will only take 4KB to 8KB of upper memory. When the device boots, the PXE runtime code is copied from the boot PROM into the top of free base memory.

E100_S.NIC and E100_S.LOM are designed to run from the NIC boot PROM and system BIOS boot PROM, respectively. These PXE ROM images do not contain PXE BaseCode. PXE BaseCode must be implemented separately. The files (PXEBASE.NIC and PXEBASE.LOM) are provided for this purpose. PXEBASE.LOM is to be included in the BIOS boot PROM. PXEBASE.NIC is to be programmed into a second NIC boot PROM (this is for initial testing before PXEBASE.LOM is programmed into the BIOS boot PROM).

Q2. Why does the BIOS lock up during option ROM scan after running E100_S.LOM?

A2. Some BIOS's are not restoring the correct Global Descriptor Table (GDT) when the Protected Mode Copy (Int 15h, AH=87h) service is run during option ROM scan. These BIOS's are usually in a flat memory model during option ROM scan, and the Protected Mode Copy service returns with the processor in real-mode.

You can use the NIC version (E100_M.NIC or E100_S.NIC) to test the rest of the PXE 2.0 code, using a NIC, with a BIOS that has the Protected Mode Copy bug. To fix this bug, the BIOS needs to restore the correct GDT while in option ROM scan.

Q3. How is PXE 2.0 PROM selected to be the boot device?

A3. The PXE 2.0 is a standard PCI/PnP option ROM.

If the PXE 2.0 is placed into a BIOS that supports PnP/BBS (BIOS Boot Specification), the BIOS should insert a network boot device into the boot order list.

If the PXE 2.0 is placed into a BIOS that does not support PnP/BBS, that BIOS must support network devices that hook interrupt vector 18h. After the PXE 2.0 returns control to the BIOS (at the end of the option ROM initialization call), the BIOS should check to see if Int 18h has changed. If it has, the BIOS should assume that a network boot device has hooked Int 18h and give the user the ability to select network boot in the CMOS setup. Normally, a BIOS gives the user the ability to boot drive A:, C:, and the CD-ROM drive. It now needs to add network to the list of boot devices.

Q4. Our BIOS supports PnP/BBS. Why does the PXE 2.0 still hook bootstrap interrupt 18h (or 19h)?

A4. The EEPROM on the NIC (or LOM) needs to be reprogrammed. Use the EEPROM.EXE utility included in this PDK to change the NIC bootstrap selection.

If BBS is selected in the EEPROM and the NIC will still not boot, please verify that:
1. The PnP installation check structure (in the system BIOS) is paragraph (16 byte) aligned between addresses E0000h and FFFF0h.
2. The PnP installation check structure is valid (checksum and all fields are correct) during option ROM initialization.

Q5. Does PXE 2.0 use POST Memory Manager if it is supported in the system BIOS?

A5. Yes, the LOM version (E100_S.LOM) will use POST Memory Manager if it is available. PXE 2.0 will allocate a 64KB block using POST Memory Manager. If this allocation fails, it will leave the entire ROM image in upper memory.

Q6. Why does my DOS boot image hang when I use HIMEM.SYS?

A6. For HIMEM.SYS, you must use /TESTMEM:OFF. If this parameter is not used, HIMEM.SYS will overwrite the DOS RAMDISK image that was stored in extended memory by the PXE bootstrap program.

Q7. Why does the Microsoft DOS TCP/IP hang when booting with BOOTP?

A7. Be sure to initialize these vendor options: IP Address Lease Time (51), Rebinding Time Value (59) and Renewal Time Value (58) to some value besides 'infinite' (all ones). Also, do not exceed a maximum packet size of 512 bytes.

Q8. Why does multicast discovery and multicast TFTP not work with Microsoft NT software router?

A8. Microsoft router does not seem to support multicast packet forwarding (at least it was not obvious to us). If you must use this router, turn off multicast TFTP and multicast discovery.

Q9. I installed the PXE PDK on my Microsoft DHCP server (that also has the NT software router). I disabled multicast and the client still will not boot.

A9. You cannot use broadcast discovery when installing the PDK on your DHCP server. You cannot use multicast with a Microsoft router. In this case, you can only use unicast discovery. Disable broadcast discovery, disable multicast discovery and add the PXE server IP address to the discovery list.

Q10. Do I need to worry about the PXENV+ structure or just the !PXE structure when writing my own UNDIs/BaseCode/NBPs/protocol drivers?

A10. Yes. It is very important, for backward compatibility, to only reference the PXENV+ structure in the UNDI, so your UNDI will work with existing WfM-1.1a compatible NBPs and protocol drivers. It is also very important to update the !PXE structure for future compatibility (the PXENV+ will be removed in a future revision of the PXE specification). For now, always search for, and update, the !PXE and PXENV+ structures when any changes need to be made.

Q11. When using the TFTP API, if I open a TFTP transaction with a bigger packet size on a non standard TFTP port why does my client not accept the data packets?

A11. There is a problem in the existing TFTP code (build 68 or less) which mistakes any non-standard TFTP port for an MTFTP port. In an MTFTP transaction you don't ask for bigger packets but simply accept the packet size that is coming in. The workaround for this is to give 512 as the packet size if you are using a non-standard TFTP port. If the server can send bigger packets, the client will be able to accept them and adjust the packet size accordingly.

Q12. Should my NBP use the !PXE or PXENV+ structure?

A12. NBPs should always use the !PXE structure, if present. The PXENV+ structure is going to be removed in a future release of the PXE specification.

Q13. How should my NBP search for the !PXE and PXENV+ structures?

A13. NBPs should check for !PXE and PXENV+ structures in the following order:

1. !PXE structure address on CPU stack
2. !PXE structure address in DX:AX (seg:off)
3. !PXE structure via memory scan
4. PXENV+ structure in ES:BX
5. PXENV+ structure via Int 1Ah AX=5650h
6. PXENV+ structure via memory scan

If PXENV+ is found, check for the address of the !PXE structure at end of PXENV+ structure. NBPs should use the !PXE structure API entry point if it is available. The PXENV+ structure will be removed in a future release of the PXE specification.

Q14. Why do I need Windows NT 4 Server CD to install the PDK on Windows 2000 Server?

A14. The PDK needs the DOS network client files from the NT 4 Server CD. These files are not included on the Windows 2000 Server CD.